THE UNIVERSITY
of EDINBURGH

Efficient human annotation schemes for training object class detectors

Dimitrios P. Papadopoulos

Doctor of Philosophy

Institute of Perception, Action and Behaviour

School of Informatics

University of Edinburgh

2018

Abstract

A central task in computer vision is detecting object classes such as cars and horses in complex scenes. Training an object class detector typically requires a large set of images labeled with tight bounding boxes around every object instance. Obtaining such data requires human annotation, which is very expensive and time consuming. Alternatively, researchers have tried to train models in a weakly supervised setting (i.e., given only image-level labels), which is much cheaper but leads to weaker detectors. In this thesis, we propose new and efficient human annotation schemes for training object class detectors that bypass the need for drawing bounding boxes and reduce the annotation cost while still obtaining high quality object detectors.

First, we propose to train object class detectors from eye tracking data. Instead of drawing tight bounding boxes, the annotators only need to look at the image and find the target object. We track the eye movements of annotators while they perform this visual search task and we propose a technique for deriving object bounding boxes from these eye fixations. To validate our idea, we augment an existing object detection dataset with eye tracking data.

Second, we propose a scheme for training object class detectors, which only requires annotators to verify bounding-boxes produced automatically by the learning algorithm. Our scheme introduces human verification as a new step into a standard weakly supervised framework which typically iterates between re-training object detectors and re-localizing objects in the training images. We use the verification signal to improve both re-training and re-localization.

Third, we propose another scheme where annotators are asked to click on the center of an imaginary bounding box, which tightly encloses the object. We then incorporate these clicks into a weakly supervised object localization technique, to jointly localize object bounding boxes over all training images. Both our center-clicking and human verification schemes deliver detectors performing almost as well as those trained in a fully supervised setting.

Finally, we propose extreme clicking. We ask the annotator to click on four physical points on the object: the top, bottom, left- and right-most points. This task is more natural than the traditional way of drawing boxes and these points are easy to find. Our experiments show that annotating objects with extreme clicking is $5\times$ faster than the traditional way of drawing boxes and it leads to boxes of the same quality as the original ground-truth drawn the traditional way. Moreover, we use the resulting extreme points to obtain more accurate segmentations than those derived from bounding boxes.

Acknowledgements

First and foremost, I want to thank my advisors Vittorio Ferrari and Frank Keller for all the guidance and support during my PhD. They both were true teachers for me. They taught me how I should approach an idea, how to tackle with all the difficulties I was facing and how I should properly present this idea. More importantly, they taught me what it really means to do proper research. Their passion, motivation and inspiration were constantly helping me to move forward and become a better researcher. I consider myself very fortunate that had the opportunity to work with these two wonderful people.

During my PhD, I have been very fortunate to collaborate and with Jasper Uijlings. Jasper is an excellent researcher who played the role of my third advisor. Jasper's feedback and insight helped me grow as a researcher. I am really grateful to him for his guidance and support. I would also like to thank Alasdair Clarke for helping me during the first months of my PhD and transferring to me his knowledge about eye tracking data.

I would also like to thank Timothy Hospedales and Serge Belongie for accepting to review my thesis. Moreover, thanks to the School of Informatics at the University of Edinburgh and to the ERC Starting Grant VisCul for supporting my research.

During my PhD, I had the opportunity to interact with great people from CALVIN group. My gratitudes to all CALVINees: Holger Caesar, Abel Gonzalez-Garcia, Paul Henderson, Vicky Kalogeiton, Buyu Liu, Davide Modolo, Anestis Papazoglou, Luca Del Pero, Miaojing Shi, Jasper Uijlings, Alexander Vezhnevets, Michele Volpi for the all the interesting discussions and also for all the fun nights out.

I would also like to thank my friends Vicky Kalogeiton, Stavros Gerakaris, Juan Fumero, Stratos Samaridis, Abel Gonzalez-Garcia, Holger Caesar and Chris Margiolas for their support and for making my everyday life much better. They were constantly there when I wanted somebody to grumble about my problems during my PhD or when I wanted to relax from the PhD work by doing something fun. Also, special thanks goes to "Mr. Jamaica" who was the first person that was saying good morning to me after the long working nights especially before the deadlines.

Finally, I am more than grateful to my parents and my sister for their unconditional support and patience through the PhD and for helping me to pursue my dreams.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Dimitrios P. Papadopoulos*)

To those I lost,
to those I will gain.

———

To my beloved grandma (1925-2016)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

The human brain can process visual information and analyze the surrounding environment successfully. By quickly looking at a scene, we are able to recognize and deeply analyze the visual concepts of an image or a video. Even from a very young age, humans are remarkably capable of quickly detecting diverse object instances, localizing them in scenes by understanding their exact layout, understanding the relations between them, and inferring any actions that take place in a very complex scene. More importantly, humans have the ability to infer the motivations of other people's actions or predict accurately how a scene will look in the immediate future.

Albeit trivial even for three-year old children, these tasks are extremely challenging even for our most advanced machines. Computer Vision is the field of Artificial Intelligence that tries to solve all these tasks. The grand goal of computer vision is to make computers see and understand the visual content of an image or a video in the same way humans do. Until recently (2012), our computers struggled at performing even close to human performance at any of the aforementioned tasks. During the last five years (2012-2017), with the evolution of deep learning and of the powerful Convolutional Neural Networks (CNNs) [Krizhevsky et al., 2012], we observed a dramatic increase in performance on various important visual recognition tasks, such as image classification, object detection or action recognition. In 2015, the first computer vision

algorithm seemed to perform similarly with humans in the task of image classification on the popular ImageNet image classification challenge [Russakovsky et al., 2015a]. Nevertheless, there is still a long way to go before our machines understand the world around us.

A fundamental component of most computer vision algorithms is the learning procedure. Similar to humans, the machine needs to be trained beforehand with a lot of examples in training images in order to be able to learn a visual model and make prediction in new images. Obtaining such data requires human annotation which is very expensive and time consuming. This thesis addresses this annotation problem in the object detection task. Our goal is to propose efficient annotation schemes that reduce the manual human annotation cost.

Outline. The problem statement of this thesis is presented in Section 1.1. In Section 1.2, we describe the main contributions of this thesis, while in Section 1.3 we present the list of the publications resulting from these contributions. Finally, in Section 1.4 we present how the thesis is organized.

1.1 Problem statement

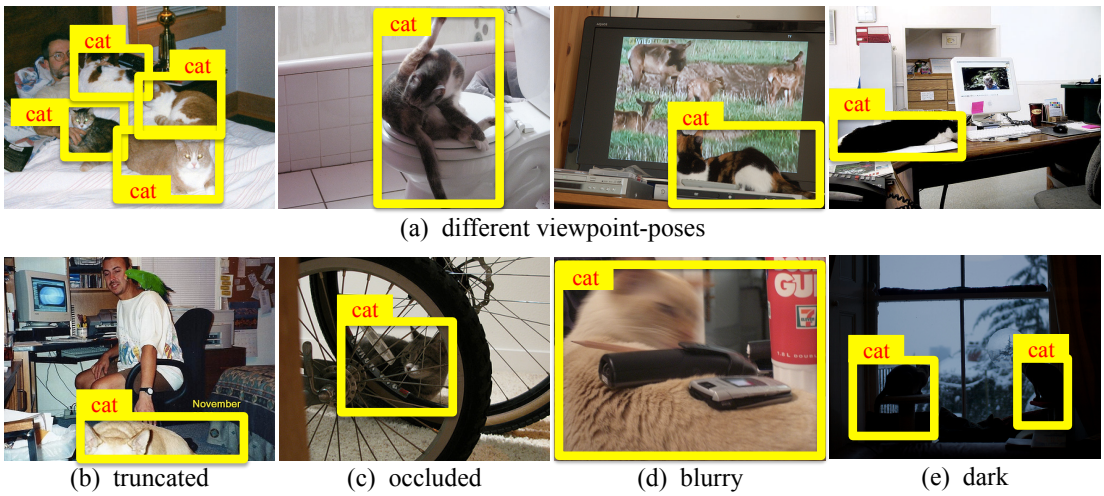
Detecting objects in images is fundamental in computer vision. Object class detection is the task of predicting a bounding box around each instance of a given object class in a test image (Figure 1.1). Detecting and localizing objects in real-world images is a highly challenging task (Figure 1.2). The appearance of an instance may differ significantly from other instances of the same class (intra-class variation). Objects can be found in a wide variety of different poses and viewpoints, which can significantly change their visual appearance. Also, objects may appear truncated by the image border or partially occluded by other objects.

Training object class detectors typically requires a large, diverse set of images in which objects are manually annotated with tight bounding boxes drawn by humans (full supervision, Figure 1.3). This task is tedious, very time consuming and expensive. For instance, annotating ILSVRC [Russakovsky et al., 2015a], currently the most popular object class detection dataset, required 35s per bounding-box by crowd-sourcing on Mechanical Turk using a technique specifically developed for efficient bounding-box annotation [Su et al., 2012].

This enormous annotation cost has been the main driving factor that led the com-

Figure 1.1: *Given an image of a scene, an object class detector should predict a bounding-box around each instance of an object class. The figure presents object detection predictions for the motorbike class.*

puter vision community to explore alternative annotation methods. Weakly Supervised Object Localization (WSOL) has become a significant task in recent years to reduce the manual labeling effort to learn object classes [Fergus et al., 2003; Chum and Zisserman, 2007; Nguyen et al., 2009; Deselaers et al., 2010; Pandey and Lazebnik, 2011; Siva and Xiang, 2011; Russakovsky et al., 2012; Siva et al., 2012; Shi et al., 2013; Song et al., 2014a,b; Bilen et al., 2014; Cinbis et al., 2014; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016]. These methods are trained from a set of images labeled only as containing a certain object class, without being given the location of the objects (weak supervision, Figure 1.3). The goal of a WSOL method is to localize the objects in its own training images while learning an object detector for localizing instances in new test images. While the weakly supervised setting is substantially cheaper, the resulting object detectors typically deliver only about half the accuracy of their fully supervised counterparts [Deselaers et al., 2010; Siva and Xiang, 2011; Siva et al., 2012; Shi et al., 2013; Song et al., 2014a,b; Bilen et al., 2014; Cinbis et al., 2014; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016].



(a) different viewpoint-poses

(b) truncated (c) occluded (d) blurry (e) dark

Figure 1.2: ***Challenges in object detection.*** *Objects may be found in a wide variety of (a) different poses and viewpoints, may appear (b) truncated by the image border or (c) partially occluded by other objects or may be (d) very blurry or very dark. All these factors can significantly change the visual appearance of the objects making the object detection task very challenging. The figure presents examples from the COCO dataset [Lin et al., 2014] for the cat class.*

1.2 Contributions

In this thesis we propose four new and efficient human annotation schemes for training object class detectors. Our schemes require minimal human supervision, reducing the enormous annotation cost of drawing bounding boxes. More particularly, the contributions of this thesis are:

- A novel approach to training object detectors from eye tracking data. Instead of carefully marking every training image with accurate bounding-boxes, the annotators only need to find the target object in the image, look at it, and press a button. By tracking the eye movements of the annotators while they perform this task, we obtain valuable information about the position and size of the target object. This task can be performed very efficiently by human observers, especially compared to the time required to draw a bounding box. To test our hypothesis that eye tracking data can reduce the annotation effort required to train object class detectors, we collected eye tracking data for the popular PASCAL VOC 2012 [Everingham et al., 2012]. We then propose a technique for deriving object bounding-boxes from these eye fixations and we show that any standard object class detector can be trained on the bounding-boxes predicted by our model

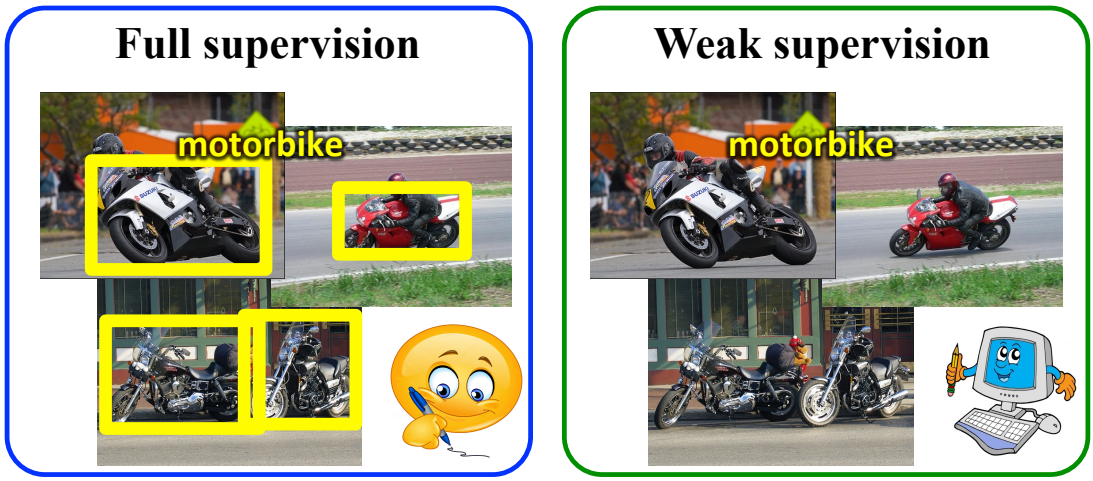


Figure 1.3: *Object class detectors are typically trained under full supervision (left): a large, diverse set of images in which objects are manually annotated with tight bounding boxes is required. In the weakly supervised setting (right), the detector is trained from a set of images labeled only as containing a certain object class. In contrast to full supervision, the location annotation of the objects is not given.*

(Chapter 3).

- An efficient annotation scheme for training object detectors which only requires annotators to verify bounding-boxes produced automatically by the learning algorithm. Our scheme iterates between re-training the detector, re-localizing objects in the training images, and human verification. We use the verification signal both to improve re-training and to reduce the search space for re-localization, which makes these steps different to what is normally done in a weakly supervised setting. In extensive experiments on PASCAL VOC 2007 [Everingham et al., 2007], we show that (1) using human verification to update detectors and reduce the search space leads to the rapid production of high-quality bounding box annotations; (2) our scheme delivers detectors performing almost as good as those trained in a fully supervised setting, without ever drawing any bounding boxes; (3) as the verification task is much faster than drawing, our scheme substantially reduces total annotation time (Chapter 4).

- An efficient annotation scheme for training object detectors with center-click supervision. We ask annotators to click on the center of an imaginary bounding box which tightly encloses the object instance. We then incorporate these clicks into existing Multiple Instance Learning techniques for weakly supervised object

localization, to jointly localize object bounding boxes over all training images. In extensive experiments on PASCAL VOC 2007 [Everingham et al., 2007] and MS COCO [Lin et al., 2014], we show that (1) our scheme delivers high-quality detectors, performing substantially better than those produced by weakly supervised techniques, with a modest extra annotation effort; (2) these detectors in fact perform in a range close to those trained from manually drawn bounding boxes; (3) as the center-click task is very fast, our scheme substantially reduces total annotation time (Chapter 5).

- Extreme clicking for efficient object annotation. The traditional way of drawing bounding boxes [Su et al., 2012] is very time consuming and involves clicking on imaginary corners of a tight box around the object. This is difficult as these corners are often outside the actual object and several adjustments are required to obtain a tight box. Instead, we ask the annotator to click on four physical points on the object: the top, bottom, left- and right-most points. This task is more natural and these points are easy to find. We crowd-source extreme point annotations for PASCAL VOC 2007 [Everingham et al., 2007] and 2012 [Everingham et al., 2012] and show that (1) annotation time is only 7s per box, 5$\times$ faster than the traditional way of drawing boxes [Su et al., 2012]; (2) the quality of the boxes is as good as the original PASCAL ground-truth drawn the traditional way; (3) detectors trained on our annotations are as accurate as those trained on the original ground-truth. Moreover, extreme clicking not only yields box coordinates, but also four accurate boundary points. We show (4) how to incorporate them into GrabCut [Rother et al., 2004] to obtain more accurate segmentations than those delivered when initializing it from bounding boxes; (5) semantic segmentations models trained on these segmentations outperform those trained on segmentations derived from bounding boxes (Chapter 6).

1.3 Publications

The contributions presented in this thesis have been published in the following papers:

- D.P. Papadopoulos, A.D.F. Clarke, F. Keller, and V. Ferrari. Training object class detectors from eye tracking data. In *Proceedings of the European Conference on Computer Vision (**ECCV**)*, 2014.

- D.P. Papadopoulos, J.R.R. Uijlings, F. Keller, and V. Ferrari. We dont need no bounding-boxes: Training object class detectors using only human verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition **(CVPR)***, 2016. (spotlight oral)

- D.P. Papadopoulos, J.R.R. Uijlings, F. Keller, and V. Ferrari. Training object class detectors with click supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition **(CVPR)***, 2017. (spotlight oral)

- D.P. Papadopoulos, J.R.R. Uijlings, F. Keller, and V. Ferrari. Extreme clicking for efficient object annotation. In *Proceedings of the International Conference on Computer Vision **(ICCV)***, 2017.

1.4 Thesis outline

The rest of the thesis is organized as follows: Chapter 2 presents the related work. Chapter 3 presents our annotation scheme of training object detectors from eye tracking data. Chapter 4 presents our scheme of training object detectors using only human verification. Chapter 5 presents another annotation scheme where annotators are merely need to click on the center of the objects. Chapter 6 presents our extreme clicking strategy for annotating object instances. Finally, Chapter 7 briefly summarizes our contributions, draws general conclusions by comparing our schemes and presents perspectives for future work.

Chapter 2

Related work

Contents

In this chapter, we first provide an overview of the object class detection task and we briefly review the most important landmarks in the history of object detection (Section 2.1). In Section 2.2, we provide an overview of weakly supervised object localization techniques, where the task is to train object detectors from image-level labels only. Finally, in Section 2.3, we discuss other alternative ways to reduce human annotation effort for training object detectors, such as transfer learning or using other forms of supervision such as text that naturally occurs near an image.

2.1 Object class detection

Object class detection is one of the most challenging and important tasks in computer vision [Murase and Nayar, 1995; Lowe, 1999; Schmid and Mohr, 1996; Rothganger et al., 2003; Mahamud and Hebert, 2003; Tuytelaars and Van Gool, 2004; Moreels et al., 2004; Ferrari et al., 2004; Viola and Jones, 2001; Viola et al., 2005; Dalal and Triggs, 2005; Vedaldi et al., 2009; Ferrari et al., 2010; Felzenszwalb et al., 2010; Malisiewicz et al., 2011; Alexe et al., 2010; Felzenszwalb et al., 2010; Uijlings et al., 2013; Sermanet et al., 2014; Girshick et al., 2014; He et al., 2014; Girshick, 2015; Simonyan and Zisserman, 2015; Ren et al., 2015; Gidaris and Komodakis, 2016; He et al., 2016; Liu et al., 2016]. Object class detection is the task of predicting a tight axis-aligned rectangle (called bounding box) around each instance of a given object class in a test image (Figure 1.1).

Most of the object class detectors follow a common procedure: At training time, a window classifier is trained to decide whether a window contains an instance of the target object. At test time this classifier is applied to score windows on a test image. The local maxima of the score function are returned as the detections. Generally, image windows are represented with robust feature vectors. After the arrival of Convolutional Neural Nets (CNNs) in computer vision, CNN-based object detectors are training in an end-to-end fashion by learning the feature representation together with the window classifier instead of using hand-crafted features to represent the visual content of the image windows. Over the last two decades, several object class detectors have been proposed. The most commonly used modern object detectors are: R-CNN [Girshick et al., 2014], OverFeat [Sermanet et al., 2014], SPP [He et al., 2014], MRCNN [Gidaris and Komodakis, 2015], Fast R-CNN [Girshick, 2015], Faster R-CNN [Ren et al., 2015], YOLO [Redmon et al., 2016], SSD [Liu et al., 2016] and YOLO9000 [Redmon and Farhadi, 2016].

In this Section, we first present an overview of the most important landmarks in the history of object detection (Sections 2.1.1-2.1.4) and especially we focus on the detectors used in the experiments presented in this thesis [Felzenszwalb et al., 2010; Girshick et al., 2014; Girshick, 2015]. Then, we briefly review the evolution of the object detection performance (Section 2.1.5). In Section 2.1.6 we focus on quantifying the human annotation cost needed to train these object detectors which is particularly the main concern of this thesis.

2.1.1 Interest points Era

The early efforts in object detection include appearance-based methods [Murase and Nayar, 1995; Lowe, 1999; Schmid and Mohr, 1996; Rothganger et al., 2003; Mahamud and Hebert, 2003; Tuytelaars and Van Gool, 2004; Moreels et al., 2004; Ferrari et al., 2004] that aim to model the appearance of local object patches. First, these methods detect a set of interest points in a test image, then they extract appearance features of the local patches around them and finally match them with features extracted from the training images by using some distance function. Affine-invariant interest point detectors [Matas et al., 2002; Mikolajczyk and Schmid, 2004; Tuytelaars and Van Gool, 2004], have been widely used in such methods and it is capable of matching local object regions across images taken from different viewpoints. The appearance of local patches is described with a local descriptor (e.g.,the widely used Scale-Invariant Feature Transform (SIFT) [Lowe, 1999]).

2.1.2 Sliding-window Era

After the early efforts that rely on interest points, a lot of object detections systems rely on the sliding-window paradigm [Viola and Jones, 2001; Viola et al., 2005; Dalal and Triggs, 2005; Vedaldi et al., 2009; Ferrari et al., 2010; Felzenszwalb et al., 2010; Malisiewicz et al., 2011]. As an object can be located at any position and scale in the image, these methods search exhaustively in a dense grid of the image. First, a window classifier is trained to decide whether a window contains an instance of the target object and then at test time is applied to score every window on a dense grid. The local maxima of the score function are returned as the detections.

Image windows can be represented by local features. The sliding window approaches based on Bag-of-words (BoW) [Sivic and Zisserman, 2003a; Csurka et al., 2004a; Vedaldi et al., 2009; Harzallah et al., 2009] represent an image window as an orderless collection of local features. First, these methods extract local features (e.g.,SIFT [Lowe, 1999] or SURF [Bay et al., 2008] around points. Typically, these points are not detected by a keypoint detector but selected from a dense image grid. Then the extracted features are clustered and get quantized into visual words (clusters). The set of visual words form the visual codebook. Finally, an image region is represented by a histogram of visual words. These histograms are then used to train complex classifiers (e.g., x^2-SVM [Maji et al., 2008], EMD-SVM [Zhang et al., 2007]). Spatial pyramid matching (SPM) models [Lazebnik et al., 2006; Uijlings et al., 2013] improve

BoW representations by augmenting it with global spatial relations.

Dalal and Triggs [2005] proposed a sliding window detector for pedestrian detection. They introduced a very successful feature descriptor called Histogram of Gradients (HOG). HOG descriptor is very similar to SIFT [Lowe, 1999] but it also does contrast normalization over small overlapping cells As window classifier, they use an SVM. An interesting component of their training procedure is the hard-negative mining. An initial model is trained in order to classify negative windows on training images. Incorrectly classified negative windows (hard-negatives) are collected and are used to train a new model. This process may be repeated a few times until the model converges.

DPM detector. Besides the above approaches, a widely used object class detector is the Deformable Part Model detector (DPM) [Felzenszwalb et al., 2010]. This object detection system represents highly variable object classes using mixtures of multi-scale deformable part models. DPM is built on the pictorial structures framework [Felzenszwalb and Huttenlocher, 2005; Fischler and Elschlager, 1973]). Pictorial structures represent objects by a collection of parts arranged in a deformable configuration. Each part captures local appearance properties of an object. The deformable configuration captures the connections between the parts using a star-structured part based model defined by a root HOG [Dalal and Triggs, 2005] filter and a set of part HOG filters. The root filter covers the entire object, while the part filters cover smaller parts of the object. An object class is actually represented by a mixture of star models, each one captioning a different viewpoint of the object. Figure 2.1 depicts a mixture model with 2 components for the bicycle class.

In Section 3.5, we train DPM detectors on bounding boxes obtained by our proposed scheme and we compare their performance to the one of DPM detectors trained on ground-truth bounding boxes.

2.1.3 Object proposals Era

The high number of windows need to be evaluated pushed sliding window detector design towards window classifiers that can be evaluated very quickly. Proposal-based object detectors aim initially to generate for each image a small number of windows likely to contain all objects of the image regardless their class. Then, unlike exhaustive search of sliding window approaches, the class-specific window classifier is evaluated only at this limited number of proposed windows. For this reason, object proposal

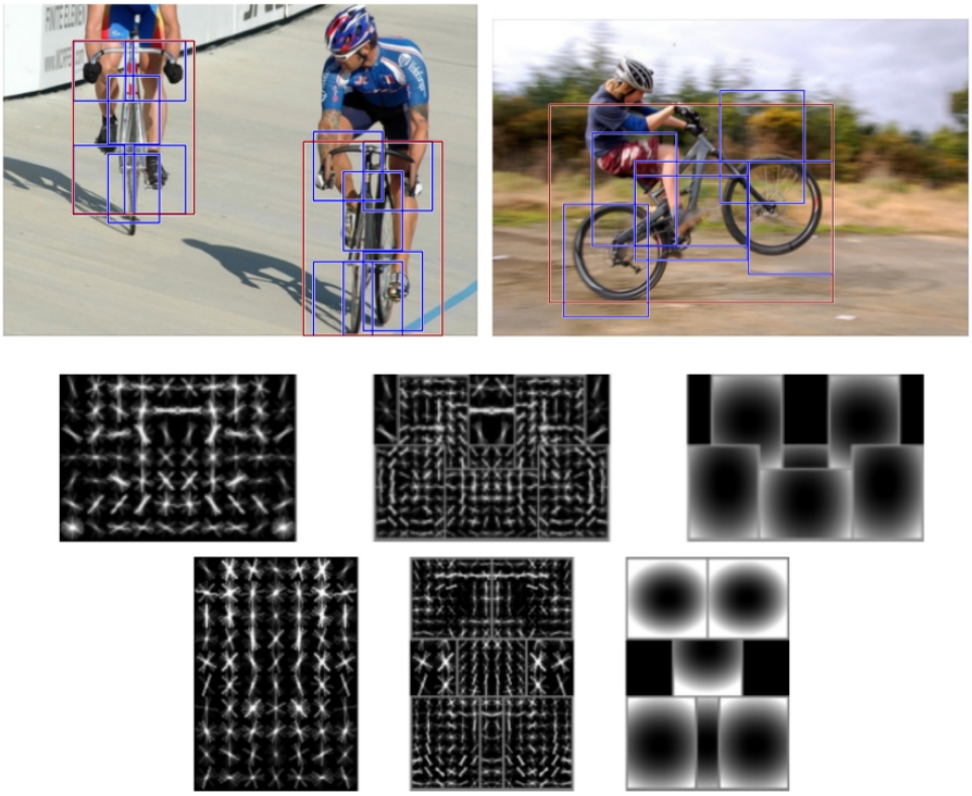


Figure 2.1: ***DPM detector:*** *Detections obtained with a two component bicycle model. The first component captures sideways views of bicycles while the second one frontal and near frontal views (Image source [Felzenszwalb et al., 2010]).*

techniques opened a new horizon of building much heavier window classifiers.

Among the very first object proposal generators [Alexe et al., 2010; Carreira and Sminchisescu, 2010; Endres and Hoiem, 2010], Alexe et al. [2010] presented a generic objectness measure that quantifies how likely it is for a window to contain an object of any class. The objectness measure can be used as location prior to any sliding window detector in order to greatly reduce the number of windows evaluated (10×-40× fewer windows). Interestingly, they show that this massive reduction in the numbers of windows evaluated using objectness on three object detectors [Dalal and Triggs, 2005; Felzenszwalb et al., 2010; Vedaldi et al., 2009] comes with little compromise on performance. Since 2010, many methods that generate a set of class-generic object proposals have been proposed [Rahtu et al., 2011; Van de Sande et al., 2011; Alexe et al., 2012; Carreira and Sminchisescu, 2012; Ristin et al., 2012; Uijlings et al., 2013; Manen et al., 2013; Arbeláez et al., 2014; Cheng et al., 2014; Rantalankila et al., 2014; Krähenbühl and Koltun, 2014; Oneata et al., 2014; Zitnick and Dollár, 2014; Sun and Batra, 2015].

In Uijlings et al. [2013], their selective search strategy uses a hierarchical grouping to deal with all possible object scales. They increase diversity by using a variety of grouping strategies and a variety of color spaces to deal with different invariance properties and a variety of region-based similarity functions to deal with the diverse nature of objects. Also, they showed that the limited number of object proposals makes it feasible to use a strong window classifier based on dense BoW features that are very expensive to compute. Their BoW approach was the winner of the PASCAL VOC2012 detection challenge [Everingham et al., 2012]. Until today, selective search is among the most widely used strategy to generate window proposals.

In this thesis, we mostly use EdgeBoxes [Zitnick and Dollár, 2014] which gives us an "objectness" measure [Alexe et al., 2010]. EdgeBoxes [Zitnick and Dollár, 2014] generates object proposals directly from edges. Their core idea is that the number of contours wholly enclosed by a window proposal is indicative of its likelihood to contain an object. A contour is wholly enclosed by a window proposal if all edge pixels belonging to the contour lie within the interior of the window. They simply propose to score every proposal based on the number of contours it wholly encloses.

2.1.4 Deep learning Era

Convolutional Neural Networks (CNNs) have been heavily used in 1990, since they constitute a class of models with a large learning capacity. Initially, these models had several limitations. They require large amounts of training data and they also tend to easily overfit because of the the huge number of the model parameters.

The most important landmark of CNN was in 2012, when Krizhevsky et al. [2012] proposed to use CNNs for image classification. They trained a large deep CNN on 1.2 million labeled images from ImageNet dataset [Russakovsky et al., 2015a] and added a new regularization technique (dropout) that reduce overfitting. They achieved impressive results on the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Russakovsky et al., 2015a]. After this big success, the state-of-the-art object class detectors have heavily relied on CNNs and have achieved impressive results [Sermanet et al., 2014; Girshick et al., 2014; He et al., 2014; Girshick, 2015; Ren et al., 2015; He et al., 2016]. In contract to all the object detectors mentioned in the previous sections, CNN-based detectors are training in an end-to-end fashion by learning the feature representation together with the window classifier instead of using hand-crafted features to represent the visual content of the image windows.

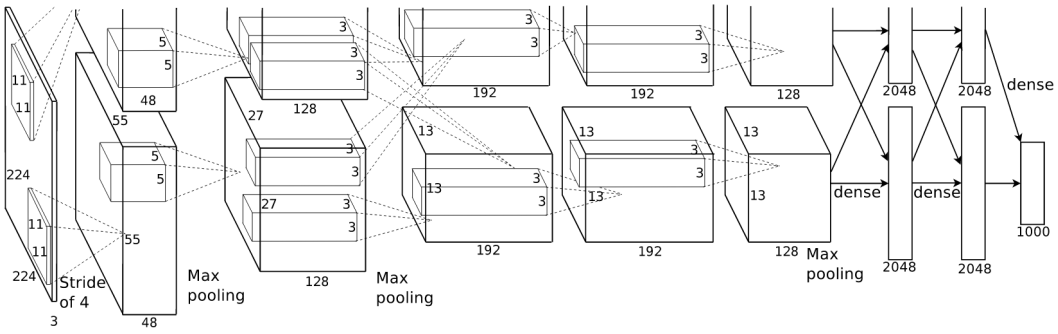


Figure 2.2: *The AlexNet architecture proposed by Krizhevsky et al. [2012] (Image source Krizhevsky et al. [2012]).*

CNNs are feed-forward neural networks that consist of a sequence of layers of neurons stacked together. Each layer transforms one set of activations to another one through a differentiable function. CNNs take as input an image and their layers transform the input to an output set through a series of hidden layers. The main type of these layers is the convolutional layer which applies a linear transformation (convolution) to its input. This linear transformation is followed by a non-linearity (e.g., REctified Linear Unit (ReLU), sigmoid). The network may also contain other types of layers, such as polling layers that are used to downsample the input or fully-connected layers. The output of the CNN is the output of the last network layer. Typically a loss function is defined on the output layer. This is usually optimized using Stochastic Gradient Descent (SGD) with backpropagation [LeCun et al., 1998]. As in the standard neural networks, backpropagation updates the weights of each layer of the network by calculating the error at the output and distributing back though the network layers by computing derivatives.

AlexNet. Figure 2.2 illustrates AlexNet, the well-known CNN architecture proposed by Krizhevsky et al. [2012]. It consists of five convolutional layers followed by three fully-connected layers. The network was initially used for image classification with 1,000 classes. In between the main eight layers there are also pooling layers and Rectified Linear Units (ReLU). Krizhevsky et al. [2012] found that using ReLU as the non-linearity decreases the training time compared to sigmoid and tanh non-linearities. To prevent overfitting, dropout layers are also implemented. During training, Krizhevsky et al. [2012] use batch Stochastic Gradient Descent (SGD). They also use data augmentation techniques, such as image translations, horizontal reflections, and patch extractions.

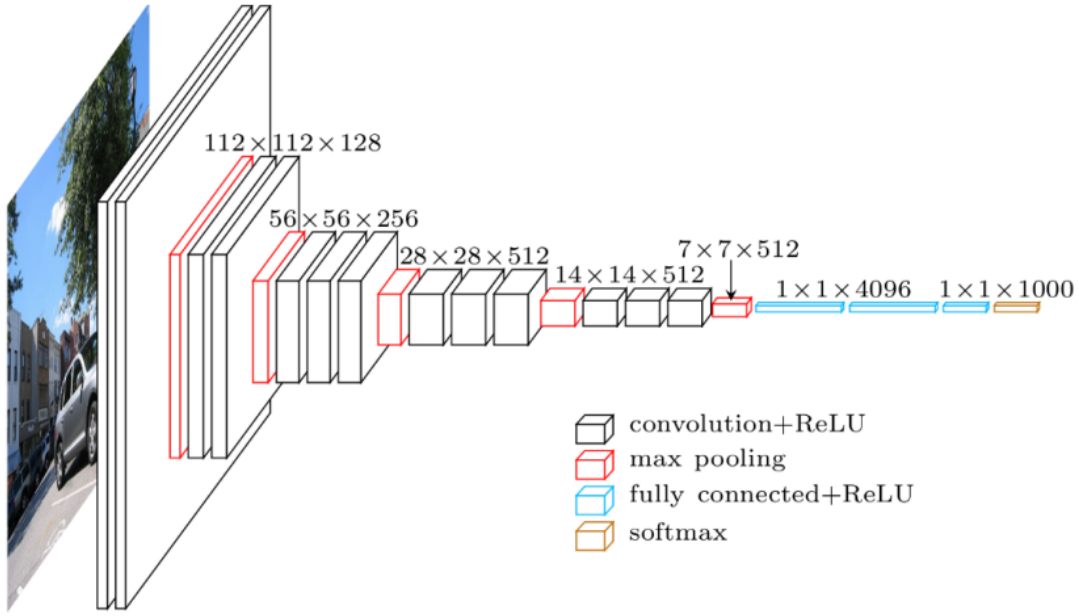


Figure 2.3: *The VGG-16 architecture proposed by Simonyan and Zisserman [2015] (Image source Noh et al. [2015]).*

Deeper CNN architectures. Depth plays an important role in the performance of the network. There is the notion that CNNs should be deep to really take advantage of the hierarchical representation of the visual data. VGG-16 [Simonyan and Zisserman, 2015] is one of the most widely used architectures as it has achieved significant performance gains in various tasks of computer vision. This network consists of 16 convolutional and fully-connected layers. The VGG-16 architecture is shown in Figure 2.3.

Residual Net (ResNet) proposed by He et al. [2016] is the current state-of-the-art network architecture, which won the ILSVRC 2015 challenge [Russakovsky et al., 2015a]. The depth of this architecture is 1,000 layers. ResNet enables these extremely deep architectures explicitly forcing its layers to learn a residual mapping by introducing skip connections between layers. It heavily uses batch normalization and unlike other architectures, it lacks of fully-connected layers at the end of the network.

Region-based Convolutional Neural Network. Region-based Convolutional Neural Network detector (R-CNN) is the first work that brought the deep learning framework into the object detection field. It was proposed by Girshick et al. [2014] and achieved impressive results on modern image datasets [Everingham et al., 2010] as it managed to improve mean average precision in object detection by more than 30%.

The R-CNN detector is shown in Figure 2.4. A window classifier based on CNNs is

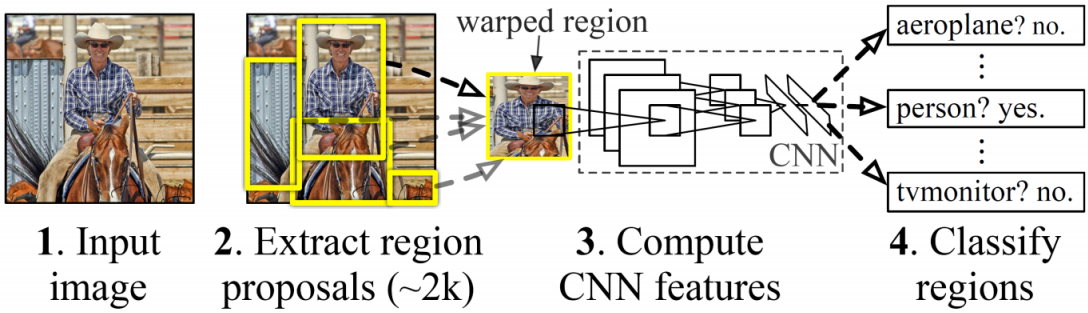


Figure 2.4: ***The R-CNN detector.*** *For a test image (1), R-CNN (2) extracts class-generic window proposals, (3) computes their CNN fixed-length features, and (4) classifies each proposal using class-specific linear SVMs. (Image source [Girshick et al., 2014]).*

applied to selective search proposals [Uijlings et al., 2013] in order to localize objects. It takes an input image and extracts selective search proposals. For each proposal, the model then computes a fixed-length feature [Krizhevsky et al., 2012; Jia, 2013] using CNN. Finally, it classifies each proposal using class-specific linear Support Vector Machines (SVMs). R-CNN also includes a per-class bounding-box regression mechanism that refines its detections to enclose the object instances more accurately.

The R-CNN detector is trained by practically fine-tuning the underlying CNN architecture which is pre-trained for image classification on the ILSVRC12 classification challenge [Krizhevsky et al., 2012]. This pre-training stage is crucial as it enables learning a discriminative visual representation by leveraging the huge amounts of annotated data that are available for this task [Russakovsky et al., 2015a].

Fast R-CNN. Fast R-CNN [Girshick, 2015] builds on the R-CNN detector and employs several innovations to improve the efficiency both at training and at test time. It processes all convolutional layers only once for the full image. After the last convolutional layer, a Region of Interest (RoI) pooling layer is introduced which extracts a fixed-length feature vector for each window proposal from the corresponding region of the convolutional map. Note that the Fast R-CNN was inspired by the SPP network [He et al., 2014], as the RoI pooling layer is a special case of the SPP layer. Fast R-CNN also introduces a multi-task loss function at the end of the network that allows training for the classification task while learning the bounding box regressor. The Fast R-CNN detector is shown in Figure 2.5

In this thesis, we mostly use Fast R-CNN detectors using AlexNet [Krizhevsky et al., 2012] or VGG-16 Simonyan and Zisserman [2015] as the underlying CNN ar-

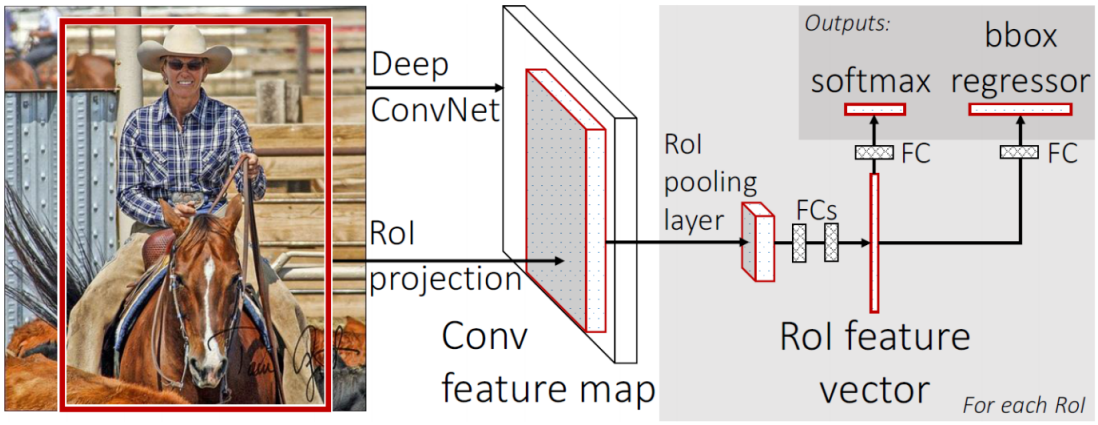


Figure 2.5: ***The Fast R-CNN detector.*** *The test image together with the window proposals are fed into the convolutional layers of the network. Each proposal is pooled and mapped to a fixed-length feature vector. The network branches into two layers, a softmax layer (classification task) and a bounding box regression layer. (Image source [Girshick, 2015]).*

chitecture.

Faster R-CNN. Faster R-CNN [Ren et al., 2015] builds upon Fast R-CNN by introducing a Region Proposal Network (RPN) that is used to generate object proposals. The RPN is placed after the last shared convolutional layer and slides over the feature map to determine whether a region is an object or not. Thus, the detector is no longer relying on external object proposal techniques that made the whole procedure slower. More importantly, the whole object detection framework can be trained end-to-end. In practice, Ren et al. [2015] propose an approximate joint training, where the RPN and the region classification network are merged into one network during training. The proposals generated by the RPN are treated as fixed by the region classification network. Faster-RCNN leads to slightly better performance than the Fast R-CNN.

2.1.5 Object detection performance

We briefly review here the evolution of performance of the most important object class detectors over the last years on the popular PASCAL VOC 2007 [Everingham et al., 2007, 2010], which consist of 20 classes. This dataset contains 5,011 trainval images and 4,952 test images. The object detectors are trained on the trainval set and we measure the object detection performance on the test set.

Evaluation. For every test image, the object detector delivers a set of object detections

(bounding boxes around the objects of each class) accompanied with the predicted detection scores. To evaluate the performance of the detector, one needs to determine whether the output detections successfully localize and recognize every object. Everingham et al. [2007] introduced Intersection-over-Union (IoU), a widely used measure to evaluate object localizations. A detection for a particular class is considered correct if the area of its intersection with any ground-truth bounding box of this class divided by the area of their union is greater than 0.5. The object detection precision is computed as the fraction of correct ($IoU > 0.5$) detections among all detections, while the recall is computed as the fraction of all instances in the test set that have been correctly detected. The Precision-Recall curve is computed by sorting all object detections by their score from which we calculate the *Average Precision* (AP) measure. In case of multiple object classes, the standard measure for evaluating object detectors is the *mean Average Precision* (mAP), which is simply the mean of the AP metrics for each class.

We now briefly review the evolution of mAP performance of the important object detectors on the PASCAL VOC 2007 dataset [Everingham et al., 2007]. In 2009, the state-of-the-art back then DPM detector achieved 29.0% mAP performance. In 2014, we had a massive jump in performance thanks to the powerful Convolutional Neural Networks (CNNs). The successful R-CNN detector of Girshick et al. [2014] achieved 58.5% mAP using the AlexNet architecture [Krizhevsky et al., 2012] and 65.4% mAP using the deeper VGG-16 architecture of Simonyan and Zisserman [2015]. Faster R-CNN detector [Ren et al., 2015] using VGG-16 yields 69.9% mAP, while using the recently proposed ResNet architecture of He et al. [2016] and more training data from COCO and VOC 2012 achieves the incredible 85.6% mAP.

2.1.6 Training object class detectors

Detecting and localizing objects in real-world images is a highly challenging task. The appearance of an instance may differ significantly from other instances of the same object class. This intra-class variation can be seen in Figure 2.6 for images of the class car. Another challenge in object detection is the wide variety of different poses and viewpoints that an object can be depicted. This can greatly affect the visual appearance. Moreover, low quality images (e.g., blurry images or low resolution images) make the task of detecting objects more difficult. Finally, detecting objects in very cluttered scenes is very challenging as objects usually appear truncated (i.e., only a part of the

Figure 2.6: *Example of intra-class variation, where instances of the same class (car) have very different appearance.*

object is visible) or partially occluded.

Typically, training object class detectors requires a large, diverse set of images in which objects are annotated with bounding boxes (full supervision, Figure 1.3). Manually drawing tight bounding boxes is very time consuming and expensive.

Time to draw a bounding box. The time required to draw a bounding box varies depending on several factors, including the desired quality of the boxes and the particular protocol used. The PASCAL VOC bounding boxes were obtained by organizing an "annotation party" where expert annotators were gathered in one place to create high quality annotations [Everingham et al., 2010].

But crowd-sourcing is essential for annotating larger sets of images with bounding boxes [Su et al., 2012; Russakovsky et al., 2015a]. A basic quality control strategy when crowd-sourcing annotations in computer vision is majority voting, i.e., collecting annotations from multiple human annotators and taking the consensus. This approach has been successfully applied to various image annotation tasks such as indicating the presence of objects or attributes [Deng et al., 2009; Sorokin and Forsyth, 2008; Lin et al., 2014]. However, the task of drawing bounding boxes is significantly more expensive than answering binary questions about the presence/absence of an object in an image.

In this thesis, as an authoritative reference we use the efficient protocol of Su et al. [2012], which was designed to produce high-quality bounding boxes with little human annotation time on Amazon Mechanical Turk. This protocol was used to annotate ILSVRC [Russakovsky et al., 2015a]. They proposed to control quality by having one

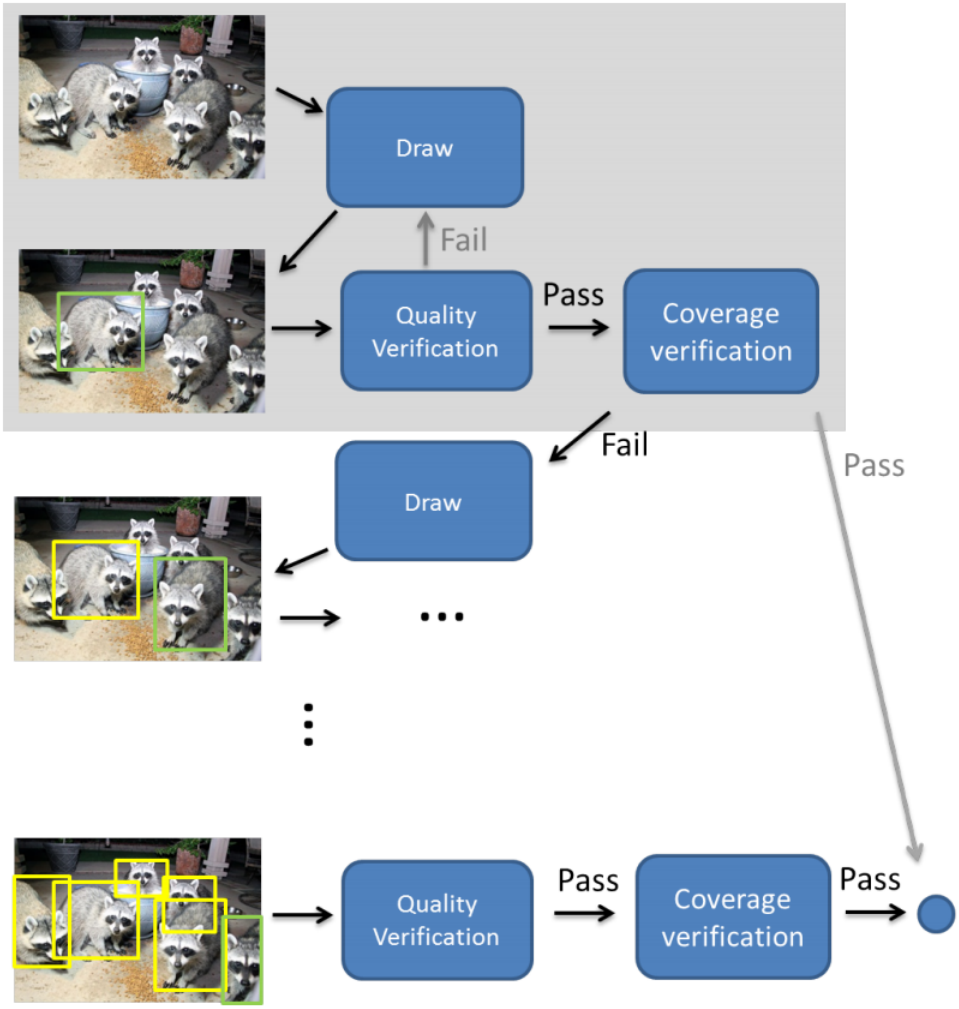


Figure 2.7: *The work flow of Su et al. [2012] for crowd-sourcing high-quality bounding box annotations (Image source [Su et al., 2012]).*

annotator draw the bounding box and another one to verify the quality of the box. Similarly, to guarantee coverage, they ask a third annotator to verify whether all object instances of an object class have bounding boxes. They report 39% efficiency gains over consensus-based approaches [Deng et al., 2009; Sorokin and Forsyth, 2008]. The work flow of their protocol is shown is Figure 2.7. Note how low quality bounding boxes are rejected during the quality verification task and how a new drawing task is generated again.

They report the following median times for annotating an object class in an image:

- 25.5s for drawing one box,
- 9.0s for verifying its quality and

- 7.8s for checking whether there are other objects of the same class yet to be annotated.

Throughout the whole thesis, we only consider localizing one object per class per image. Thus, we use 25.5s + 9.0s = 34.5s as the reference time for manually annotating a high-quality bounding box. Note that this is a conservative estimate: when taking into account that some boxes are rejected in the second step and need to be re-drawn multiple times until they are correct, the median time increases to 55s. If we use average times instead of medians, the cost is 117s.

Note how both PASCAL VOC 2007 and 2012, and ILSVRC have images of comparable difficulty and come with ground-truth box annotations of similar high quality [Russakovsky et al., 2015a], justifying our choice of 35s reference time. We also use the same reference time for COCO [Lin et al., 2014], as we conservatively assume the cost of drawing boxes would also be same as for ILSVRC, even though COCO is more difficult. All three datasets have high-quality ground-truth bounding box annotations, which we use as reference for comparisons to all our proposed annotation schemes (Sections 3–6).

Papers reporting faster timings [Jain and Grauman, 2013; Russakovsky et al., 2015b] aim for lower-quality boxes (e.g. the official annotator instructions of Jain and Grauman [2013] show an example box which is not tight around the object). In Chapter 6, we compare to Russakovsky et al. [2015b].

2.2 Weakly supervised object localization

This enormous annotation cost needed to train an object class detector (Section 2.1.6) has been the main driving factor that led the computer vision community to explore alternative annotation methods. Weakly supervised object localization (WSOL) has become a significant task in recent years to reduce the manual labeling effort to learn object classes [Fergus et al., 2003; Crandall and Huttenlocher, 2006; Chum and Zisserman, 2007; Galleguillos et al., 2008; Lee and Grauman, 2009; Blaschko et al., 2010; Deselaers et al., 2010; Pandey and Lazebnik, 2011; Siva and Xiang, 2011; Deselaers et al., 2012; Russakovsky et al., 2012; Siva et al., 2012; Shi et al., 2013; Song et al., 2014a,b; Bilen et al., 2014; Cinbis et al., 2014; Shi et al., 2015; Cinbis et al., 2016; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016]. WSOL methods are trained from a set of images labeled only as containing a certain object class. In

contrast to the fully supervised paradigm, the location annotation of the objects is not given (see Figure 1.3). The goal of a WSOL method is to localize the objects in the training images while learning an object detector for localizing instances in new test images. They typically try to locate the object by searching for patterns of appearance recurring across the training images.

2.2.1 From early efforts to CNN-based techniques

Early efforts on weakly supervised learning [Fergus et al., 2003; Crandall and Huttenlocher, 2006; Chum and Zisserman, 2007; Galleguillos et al., 2008; Lee and Grauman, 2009] have been only demonstrated on easy datasets such as CALTECH4 or Weizmann horses. The objects in such datasets are rather centered and occupy a large portion of the image. Also, there is little scale and viewpoint object variation and limited background clutter. There were also some methods that tried to experiment on more challenging datasets, such as ETHZ shape Classes or LabelMe, but they often reduced the difficulty of the dataset by manually providing information about the scale of the object [Lee and Grauman, 2009] or select easier subset of images [Chum and Zisserman, 2007].

The field has made significant progress [Deselaers et al., 2010; Pandey and Lazebnik, 2011; Siva and Xiang, 2011; Deselaers et al., 2012; Russakovsky et al., 2012; Siva et al., 2012; Shi et al., 2013], as several methods have tried to go beyond and experiment on more challenging datasets, such as PASCAL VOC 2007 [Everingham et al., 2007, 2010]. The first work on such a challenging dataset was that of Deselaers et al. [2010] which employed a conditional random field (CRF) to jointly localize and learn a new class from weakly supervised data. The key contribution of this method is the use of object proposals and the objectness score function in the WSOL task (see Section 2.2.3 for more details). Later, Siva and Xiang [2011] examined the role of intra-class and inter-class pairwise similarities in WSOL. Also, Siva et al. [2012] proposed a negative mining approach where for a positive image, a candidate window is selected such that the distance to its nearest neighbor among windows from negative images is maximal. In Cinbis et al. [2014] introduced a multi-folding idea in multiple instance learning (MIL) framework, which prevents the premature locked-in behavior.

Recent work on WSOL [Song et al., 2014a,b; Bilen et al., 2014; Cinbis et al., 2016; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016] has also shown remarkable progress, mainly because of the use of the powerful CNNs [Krizhevsky

et al., 2012; Simonyan and Zisserman, 2015], which greatly improve visual recognition tasks. For example, Bilen and Vedaldi [2016] proposed a weakly supervised deep architecture that modifies a CNN network to operate at the level of image regions, performing simultaneously region selection and classification.

At this point it is worth mentioning that any comparison between earlier WSOL methods and recent CNN-based WSOL techniques in not absolutely fair. CNN features which are used in WSOL are extracted from a network pre-trained for image classification on 1 million ILSVRC images with class labels. Even though the objects in these images are very large and relatively centred, the use of these image labels does not spoil the weakly supervised manner of the CNN-based WSOL methods. From an image annotation perspective, no bounding box is manually drawn in these ILSVRC images.

2.2.2 Performance of WSOL

In Section 2.1.5, we reviewed the incredible increase in performance of object class detectors trained under full supervision over the last years. We now review the evolution of the mAP performance of object detectors trained under weak supervision in the popular PASCAL VOC 2007 dataset [Everingham et al., 2007, 2010]. The results can be seen in Figure 2.8.

Initially, we observe that there is a significant performance gap between fully supervised and weakly supervised object detectors indicating that learning an object detector without location annotation is a very difficult task. Moreover, we observe that the performance of WSOL methods has significantly improved the last six years (from 14% mAP of Siva and Xiang [2011] which is based on DPM detector to 35% mAP of Bilen and Vedaldi [2016] which is based the deep VGG-16 CNN architecture. However, one should always pay attention on the ratio between weakly supervised and fully supervised object detectors. Interestingly, we observe that this ratio has been growing slower than the absolute value, and it is still only in the 60% range. Bilen and Vedaldi [2016] and Kantorov et al. [2016] show a slightly improved ratio (62% and 66%) over the previous CNN-based WSOL methods when they use the AlexNet architecture. However, the ratio goes down again to 56% when Bilen and Vedaldi [2016] use the deeper VGG-16 CNN architecture.

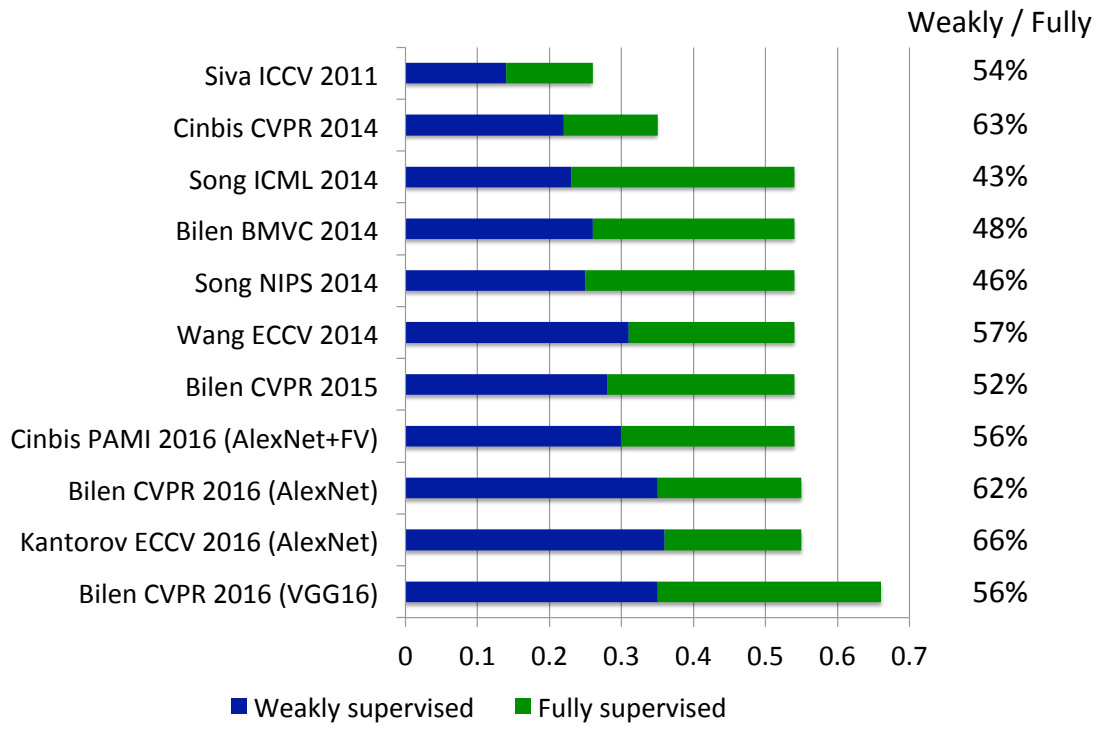


Figure 2.8: *The mAP performance of weakly supervised object detectors on the test set of PASCAL VOC 2007 (blue bars). For each method, we also provide the mAP performance of the corresponding fully supervised object detector (blue and green bars). The weakly supervised approaches produce lower quality detectors.*

2.2.3 Multiple Instance Learning

WSOL is often conceptualized as Multiple Instance Learning (MIL) [Dietterich et al., 1997; Nguyen et al., 2009; Deselaers et al., 2010; Blaschko et al., 2010; Siva and Xiang, 2011; Russakovsky et al., 2012; Siva et al., 2012; Shi et al., 2013; Song et al., 2014a,b; Bilen et al., 2014; Cinbis et al., 2014; Shi et al., 2015]. Each image is treated as a bag of windows (instances). A negative image contains only negative instances. A positive image contains at least one positive instance, mixed in with a majority of negative ones. The goal is to find the true positive instances in the positive images from which to learn a window classifier for the object class. This typically happens by iteratively alternating between two steps (Figure 2.9):

(A) re-training the object detector given the current selection of positive instances and

(B) re-localizing instances in the positive images using the current object detector.

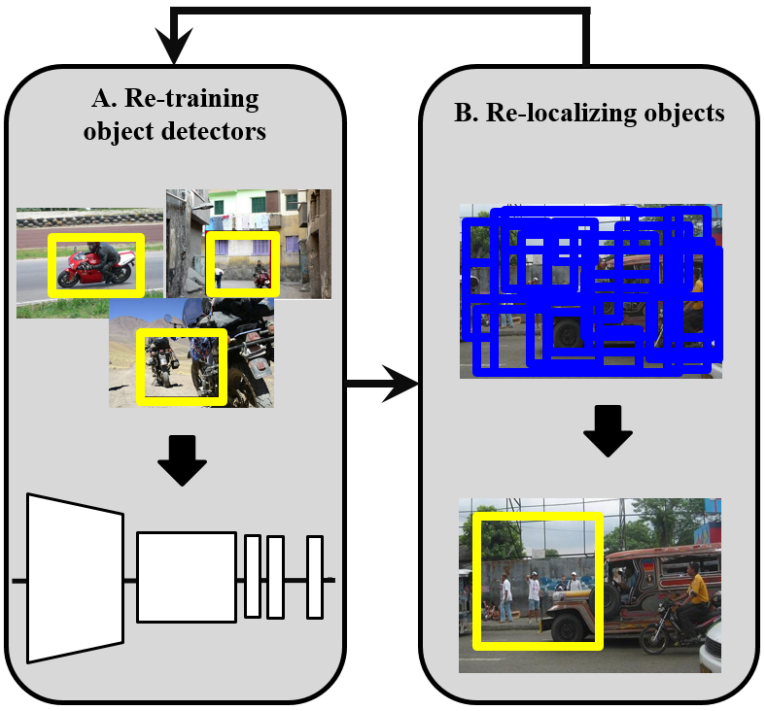


Figure 2.9: *The MIL framework widely used on WSOL. It iteratively alternates between (A) re-training the object detector given the current selection of positive instances and (B) re-localizing instances in the positive images based on the current object detector.*

Initialization. Often MIL training is initialized by taking large windows in the positive images that (nearly) cover the whole image. [Nguyen et al., 2009; Pandey and Lazebnik, 2011; Russakovsky et al., 2012; Cinbis et al., 2014, 2016]. This simple initialization strategy often works quite well on WSOL. Even in difficult datasets, such as PASCAL VOC 2007, a significant percentage of images contain big centered objects (about 17% in PASCAL VOC 2007). This strategy ensures that the first detector at the beginning of MIL is trained with some objects which are localized correctly. In literature there are also methods that use a clever way to initialize MIL [Siva and Xiang, 2011; Siva et al., 2012; Shi et al., 2013], by looking at the intra-class and inter-class pairwise similarities [Siva and Xiang, 2011] or by selecting a window that maximizes its distance to its nearest neighbor among windows from the set of the negative images [Siva et al., 2012].

Window space. As it is mentioned above, in MIL framework, each image is treated as a bag of windows. Early efforts relied on the sliding window paradigm by using a huge number of windows at all translations, scales and aspect ratios [Chum and Zisserman, 2007; Nguyen et al., 2009; Pandey and Lazebnik, 2011]. This limited the

methods to use certain classes of lightweight window classifiers, such as linear SVM on top of HoG features [Dalal and Triggs, 2005] or DPM [Felzenszwalb et al., 2010]. After Alexe et al. [2010] proposed objectness, nearly all WSOL techniques [Deselaers et al., 2010; Siva and Xiang, 2011; Russakovsky et al., 2012; Siva et al., 2012; Cinbis et al., 2014; Song et al., 2014a; Bilen et al., 2014; Bilen and Vedaldi, 2016; Kantorov et al., 2016] use only a small number of windows likely to cover all objects in the image. Object proposals made the MIL framework for WSOL more manageable and also possible to use more powerful and computational expensive models.

Re-localization. During the re-localization step (B) of MIL, the current object detector is simply applied to every window of an image. A simple yet powerful strategy is to select the window proposal with the highest detection score [Nguyen et al., 2009; Russakovsky et al., 2012; Cinbis et al., 2014, 2016]. More sophisticated re-localization strategies have also been devised, which combine the window detection score with other terms, such as exploiting the similarity between the selected windows across the positive images [Deselaers et al., 2010; Siva and Xiang, 2011; Bilen et al., 2015], selecting windows with high distance from windows in the negative images [Siva et al., 2012; Song et al., 2014a] or using more complex combinatorial criteria (e.g., min-cover in Song et al. [2014a]).

Another common component of many successful WSOL methods during the re-localization step [Deselaers et al., 2010; Siva and Xiang, 2011; Guillaumin and Ferrari, 2012; Prest et al., 2012; Shapovalova et al., 2012; Shi et al., 2012; Tang et al., 2014; Wang et al., 2014a; Cinbis et al., 2016; Bilen and Vedaldi, 2016] is the use of an objectness measure [Alexe et al., 2010; Zitnick and Dollár, 2014]. Typically, the detection score given by the object detector is combined with the objectness score which measures how likely it is that a proposal tightly encloses an object of any class (e.g. bird, car, sheep), as opposed to background (e.g. sky, water, grass). This helps the localization process to steer towards objects and away from background. The first work to introduce this idea was [Deselaers et al., 2010], using the original objectness measure of [Alexe et al., 2010]. Later works have used either that measure [Siva and Xiang, 2011; Guillaumin and Ferrari, 2012; Prest et al., 2012; Shapovalova et al., 2012; Shi et al., 2012; Tang et al., 2014; Wang et al., 2014a; Jerripothula et al., 2016] or the more modern version by Zitnick and Dollár [2014] [Cinbis et al., 2016; Bilen and Vedaldi, 2016]. In Sections 4.4, 5.4 of this thesis, we use the recent objectness measure of Zitnick and Dollár [2014].

Re-training. During the re-training step (A) of MIL, various object detection models has been used. Often, in earlier efforts [Chum and Zisserman, 2007; Nguyen et al., 2009; Deselaers et al., 2010; Russakovsky et al., 2012; Cinbis et al., 2014], a simple linear SVM is trained on standard hand-crafted features (such as HOG, Bag-of-words, etc). The DPM detector [Felzenszwalb et al., 2010] has also been utilized in the same manner Pandey and Lazebnik [2011]; Siva and Xiang [2011]; Siva et al. [2012, 2013]. Recent WSOL approaches use the powerful CNN features generated from a network pre-trained on 1 million ILSVRC images with class labels [Russakovsky et al., 2015a] and brought a massive improvement in performance [Bilen et al., 2014; Song et al., 2014a,b; Bilen et al., 2015; Cinbis et al., 2016; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016] (9 last bars of Figure 2.8).

In high dimensional feature space the discriminative SVM classifier can relatively easily separate any positive examples from negative examples, which means that most positive examples are far from the decision hyper-plane. Hence the same positive training examples used for re-training (A) are often re-localized in (B), leading to premature locked-in behavior. To prevent this, Cinbis et al. [2014] introduced multi-fold MIL: similar to cross-validation, the dataset is split into subsets, where the re-localization on each subset is done using detectors trained on the union of all other subsets. In Sections 4.4, 5.4, we adopt this multi-folding idea in our reference MIL approach.

2.2.4 Other approaches to WSOL

Non MIL. Some WSOL approaches differ from the widely used MIL framework. Most of the approaches that fall in this category are early efforts [Fergus et al., 2003; Crandall and Huttenlocher, 2006; Lee and Grauman, 2009]. Also, there are some recent WSOL approaches [Shi et al., 2012; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016] that diverge from the MIL framework and we summarize them below

Shi et al. [2012] formulates a Bayesian joint topic model on SIFT descriptors [Lowe, 2004] represented as visual words [Sivic and Zisserman, 2003b; Csurka et al., 2004b]. Each image is modeled as a distribution over object classes and background classes. By jointly modeling all classes they can ensure that two objects cannot be at the same location. Additionally, it allows sharing appearance models of the background classes.

Wang et al. [2015] select positive instances by measuring discriminativity. They first cluster windows in positive images using pLSA [Hofmann, 2001]. For each clus-

ter they create global image features using a Bag-of-Words encoding based on the N windows for each image which are closest to the cluster. With each global image feature they perform whole-image classification. Finally, they select as positive instances those in the cluster leading to the best whole-image classification.

Bilen and Vedaldi [2016] proposed an elegant weakly supervised deep CNN architecture. The core difference of this technique to MIL framework is that there is no outer loop around the detector training. All WSOL components are integrated inside the network which is trained in an end-to-end fashion. The windows are selected by a dedicated parallel detection branch of the network, which is independent of the recognition branch.

Kantorov et al. [2016] builds upon Bilen and Vedaldi [2016] and proposes a context-aware CNN architecture that exploits contextual relation between a candidate region and its surrounding regions that helps to refining the boundaries of the detected objects.

Activation maps. Recently, a different line of work in the field of weakly supervised object localization has been proposed [Simonyan et al., 2014; Oquab et al., 2015; Blot et al., 2016; Sun et al., 2016; Zhou et al., 2016; Durand et al., 2017]. In contrast to all the aforementioned WSOL approaches, these methods aim to predict only the x, y position of objects, but not their full extent (bounding boxes).

Simonyan et al. [2014] proposed to extract class saliency maps from a CNN trained for object classification and use them for WSOL. The idea is that these saliency maps encode the location of the object of the given class in the image, and thus can be used for object localisation.

Oquab et al. [2015] proposed a weakly supervised convolutional network for object classification that relies only on image-level labels. They showed that this network can learn from cluttered scenes containing multiple objects and successfully predict approximate locations, but not full extents (bounding boxes) of objects. The network outputs class activation maps for different objects. The activation maps are computed by projecting the weights of the output layer of the network to the last convolutional layers. The activation map is given by a weighted sum of the values of the feature map at the last convolutional layer. An activation map for a particular class indicates the discriminative image regions used by the CNN to identify that class. These maps are interestingly consistent with the locations of objects in the input images. The approximate location of an object is simply given by the maximal response of these output

activation maps.

Zhou et al. [2016] built on Oquab et al. [2015] and proposed to apply global average pooling instead of global max pooling to construct the activation maps. The idea is that the use of average pooling encourages the network to identify the complete extent of the object and not only one point of it.

2.3 Alternative ways to reduce annotation effort

In this section we discuss alternative types of supervision information that have been utilized to reduce human annotation effort for training object detectors, such as training object detectors from videos, transfer learning, or using other forms of supervision such as text that naturally occurs near an image.

At this point it is worth mentioning that our proposed schemes presented in Chapters 3, 4, 5, 6 provides different ways to reduce annotation effort which is complementary to all those below. Any of our schemes could potentially be integrated with some of them for even greater savings.

Learning from videos. Training object class detectors from videos could bypass the need for manual bounding boxes, as the motion of the objects facilitates their automatic localization [Kalogeiton et al., 2016; Leistner et al., 2011; Prest et al., 2012; Kumar Singh et al., 2016; Tang et al., 2013]. Leistner et al. [2011] use unlabeled video data to improve the performance of classifiers for detection on images. They detect moving objects in a video and use them. Prest et al. [2012] use weakly annotated videos, they automatically produce spatio-temporal bounding boxes (tubes) of the objects in these videos and then they used them to train object detectors. Tang et al. [2013] proposes another weakly supervised algorithm to generate spatio-temporal segments using video-level tags provided in on-line videos. Their goal is to generate video data suitable for training object detectors for image challenges.

However, because of the domain adaptation problem Kalogeiton et al. [2016], when tested on still images these detectors are still quite weak compared to ones trained on manually annotated still images.

Transfer learning. An alternative direction is transfer learning, where a model for a new class (target) is helped by labeled examples of related classes (source) Fei-Fei et al. [2007]; Lampert et al. [2009]; Salakhutdinov et al. [2011]; Aytar and Zisserman [2011, 2012]; Guillaumin and Ferrari [2012]; Kuettel et al. [2012]; Shi et al. [2012];

Vezhnevets and Ferrari [2014]; Hoffman et al. [2014]; Rochan and Wang [2015]; Uijlings et al. [2017]. These methods typically learn spatial location, appearance and context information for a new class from bounding box annotations on examples of related classes [Guillaumin and Ferrari, 2012; Shi et al., 2012; Rochan and Wang, 2015; Uijlings et al., 2017]. Kuettel et al. [2012] instead transfers segmentation masks from the source to the target examples. Hoffman et al. [2014] proposed an algorithm that transforms CNN from image classifier to an object detector without bounding-box annotated data using domain adaptation. Recently, Uijlings et al. [2017] showed that the performance of a WSOL MIL framework can be massively increased by transferring knowledge into it from object detectors trained on ground-truth bounding boxes on examples of related classes.

Learning from text. Another line of work leverages text naturally occurring near an image as a weak annotation for its contents, such as on web pages or news articles newspapers [Berg et al., 2004; Duygulu et al., 2002; Gupta and Davis, 2008; Luo et al., 2009] or even video scripts and subtitles [Everingham et al., 2006; Laptev et al., 2008; Marszalek et al., 2009; Bojanowski et al., 2013]. This form of supervision has been particularly successful for learning faces of specific people [Berg et al., 2004], in part because excellent generic face detectors are already available [Viola et al., 2005].

Chapter 3

Training object class detectors from eye tracking data

Contents

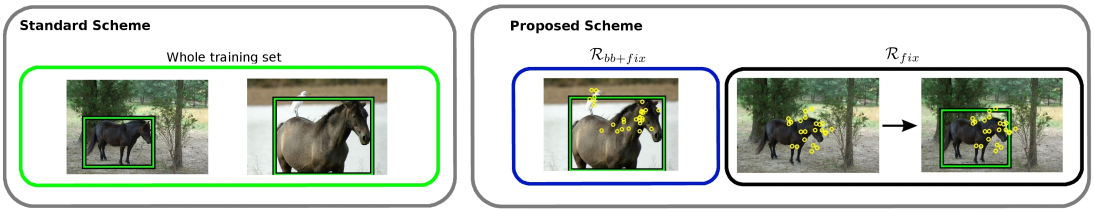


Figure 3.1: *(Left) The standard approach to training an object detector, where all images in the training set are manually annotated with bounding boxes. (Right) In our approach most of the images are annotated only with human eye fixations ($\mathcal{R}_{fix}$ set). The key idea is to automatically predict bounding boxes for the images in $\mathcal{R}_{fix}$ set given only their fixations. For this we first train a model to infer the spatial support of objects from the fixations, on a small subset of images annotated by both fixations and bounding boxes ($\mathcal{R}_{bb+fix}$, only 7% of the images in our experiments).*

3.1 Introduction

As mentioned in Section 2.1.6, training an object class detector typically requires a large set of images in which objects are manually annotated with bounding-boxes [Dalal and Triggs, 2005; Everingham et al., 2010; Felzenszwalb et al., 2010; Wang et al., 2013; Girshick et al., 2014; Girshick, 2015; Liu et al., 2016] (Figure 3.1). Bounding-box annotation is time consuming and expensive. Su et al. [2012] report a 35s median time to draw and verify a tight bounding-box during a large-scale annotation effort for ImageNet by crowd-sourcing on Mechanical Turk (more details in Section 2.1.6). Additionally, detailed annotation guidelines, annotator training based on these guidelines, and manual checking of the annotation are typically required [Everingham et al., 2010; Su et al., 2012]. Annotating large sets of images is therefore an enormous undertaking, which is typically supported by crowd sourcing [Russakovsky et al., 2015a; Everingham et al., 2012; Lin et al., 2014].

In this chapter, we propose a novel approach to training object detectors which can substantially reduce the time required to annotate images. Instead of carefully marking every training image with accurate bounding boxes, our annotators only need to find the target object in the image, look at it, and press a button. By tracking the eye movements of the annotators while they perform this task, we obtain valuable information about the position and size of the target object (Figure 3.1). Unlike bounding box annotation, our eye tracking task requires no annotation guidelines and can be carried out by completely naive viewers. Furthermore, the task can be performed in a fraction of the time it takes to draw a bounding box (about one second per image, Section 3.3).

Eye movement behavior in scene viewing has been the subject of a large body of research in cognitive science [Henderson, 2003]. Experimental results indicate that human observers prefer to fixate objects, rather than the background of an image [Einhäuser et al., 2008]. This tendency is particularly pronounced in visual search, as finding an object typically requires fixating it [Wolfe and Horowitz, 2008]. This strongly suggests that eye tracking data can be useful for training a system to automatically localize objects. However, fixation data only provides a rough indication of the spatial extent of an object: humans have a tendency to fixate the center of an object [Nuthmann and Henderson, 2010], and within animate objects (humans and animals), there is a strong preference for fixating faces [Judd et al., 2009]. Furthermore, the eye movement record will include fixations on salient non-target objects, and occasionally on the background.

Inspired by the above observations, we propose a technique for deriving a bounding box covering the whole target object given the fixations on the image (Section 3.4). We cast bounding box estimation as a figure-ground segmentation problem. We define a model that takes human eye movement data as input and infers the spatial support of the target object by labeling each pixel as either object or background. As the relation between fixations and bounding boxes can be complex and vary between object classes, we train our model from a small subset of images annotated with bounding boxes *and* fixations. The subset is only a small fraction of the dataset we want to annotate (7% of the images in our experiments). Once trained on this subset, we use our model to derive bounding boxes from eye movement data for the whole dataset, which can then be used to train any standard object class detector, such as the Deformable Parts Model [Felzenszwalb et al., 2010] or any modern detector based on Convolutional Neural Nets (CNNs), such as the Fast R-CNN detector [Girshick, 2015] or the Single Shot MultiBox Detector (SSD) [Liu et al., 2016].

To test our hypothesis that eye tracking data can reduce the annotation effort required to train object class detectors, we collected eye tracking data for the complete training set of ten objects classes from Pascal VOC 2012 [Everingham et al., 2010, 2012] (6,270 images in total). Each image is annotated with the eye movement record of five participants, whose task was to identify which object class was present in the image (Section 3.3). This large-scale eye tracking dataset is publicly available at `groups.inf.ed.ac.uk/calvin/eyetrackdataset/`.

We demonstrate through extensive experiments on this dataset that our segmentation model can produce accurate bounding boxes in about half the images, and that

a DPM object detector [Felzenszwalb et al., 2010] can be trained from them (Section 3.5).

3.2 Related Work

The work of this Chapter is related to work reducing annotation time for training object detectors. More details about WSOL or other ways to reduce annotation effort for training object class detectors can be found in Section 2.2, 2.3. In this Section we focus on related work that use eye tracking data in computer vision.

Researchers in cognitive science have a long-standing interest in computational models that predict human eye movement behavior (e.g. [Itti et al., 1998; Harel et al., 2007; Judd et al., 2009; Jiang et al., 2015; Liu et al., 2015; Kruthiventi et al., 2017]). For example, Judd et al. [2009] learn a model of saliency directly from human fixations using low, mid and high-level image features.

Recently, however, some authors have started to use eye movement data for computer vision tasks. This includes work on image segmentation, which shows that using fixation data to help segmentation algorithms leads to improved performance [Mishra et al., 2009; Ramanathan et al., 2010; Walber et al., 2013]. In Mishra et al. [2009], given a fixation used as seed point, their active image segmentation approach tries to automatically segment the region that contains this fixation point. In Ramanathan et al. [2010], they exploit the clustering of fixations around a salient object in order to automatically generate the fixation seed for the active image segmentation.

Eye tracking data has also been used for face and text detection [Karthikeyan et al., 2013]. Their approach spatially clusters the human fixations in an image in order to find regions of interests in which the targets are likely to be found, and then apply standard face and text detectors only there. Their results show that their gaze enable approach produces better results and reduces the computation time compared to the standard object detection algorithms. Several authors have collected eye tracking data for video and shown that saliency maps computed from these data can improve action recognition [Mathe and Sminchisescu, 2012; Vig et al., 2012].

Yun et al. [2013] collect eye movement data for a 1,000 image subset of Pascal VOC 2008; three observers performed a three second free-viewing task. This data is then used to re-rank the output of an object class detector [Felzenszwalb et al., 2010] on test images. Our dataset is substantially larger (6,270 images, 5 observers), and our observers perform a visual search task, which is faster (one second per image) and

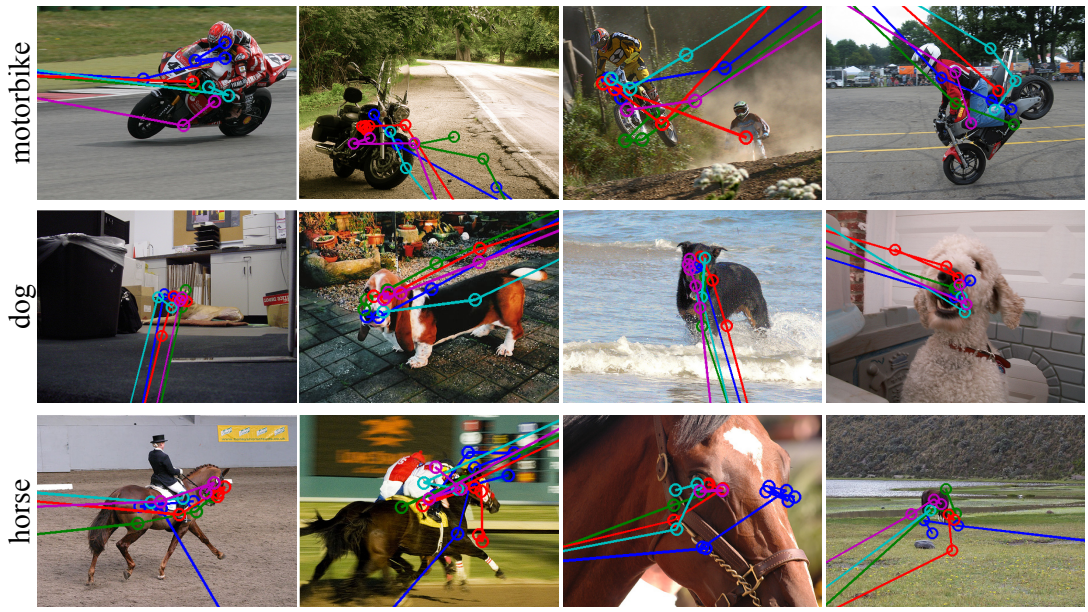


Figure 3.2: *Examples of eye tracking data for images of class motorbike, dig and horse. We show the sequence of fixations (circles) for five participants (color coded). Note how the first fixation falls outside the image (see main text for details).*

more likely to result in fixations on the target objects (Section 3.3). Most importantly, we present a method for using eye tracking data to *train* an object class detector, replacing ground-truth bounding-boxes, rather than using them for post-processing at test time.

3.3 Eye Tracking Dataset

3.3.1 Materials

The images in our dataset were taken from the 2012 edition of the Pascal VOC challenge [Everingham et al., 2012]. We selected 10 of the 20 Pascal classes and included all trainval images for these classes in our dataset. We grouped our 10 classes into pairs as follows (see below for explanation): cat/dog, bicycle/motorbike, boat/aeroplane, horse/cow, sofa/diningtable. For each pair of classes, we removed all images that contained objects of both classes (e.g. images containing both horses and cows). This eliminated 91 images, or 1.4% of total, and resulted in a dataset containing 6,270 images.

3.3.2 Procedure

As explained in Section 3.2, results in the visual cognition literature indicate that free viewing may not be the optimal task for collecting eye tracking data for training automatic object detectors. A visual search task, in contrast, increases the likelihood that participants fixate the target object, as this facilitates finding the target and help complete the task correctly.

However, traditional visual search tasks require a large number of target-absent trials. For example, if the task is to search for horses (the participants presses a "yes" button if a horse is present in the image, and "no" otherwise), then the set of images shown to the participant needs to contain images without horses (typically 50%, to minimize guessing). Such a setup would mean that eye tracking data is collected for a large number of target-absent images, which then cannot be used to train a detector.

We therefore used the related task of *two-alternative forced choice object discrimination*, in which each image contains instances of one of two object classes (e.g. cow or horse), and participants have to press one of two buttons to indicate which class is present. This way, visual search data can be collected for two classes at a time, without the need for target-absent images. In adopting this approach, we paired object classes carefully, such that the two sets of images were similar in size (to minimize guessing), and such that the objects were easily confusable (similar size and background, etc.), as otherwise the task is too easy for human observers.

3.3.3 Apparatus

The experiment was conducted in a sound-attenuated room; participants were seated 60 cm from a 22" LCD screen (Samsung SyncMaster 2233, 100 Hz refresh rate, 5 ms response time) while their eye movements were recorded using an Eyelink 2000 eye tracker (SR Research Ltd., Ottawa), which sampled both eyes at a rate of 1,000 Hz, with a typical spatial resolution of 0.25° to 0.5°. A head rest was used to minimize participants head movements and improve recording accuracy. Button presses were recorded using a Logitech gamepad that offers millisecond accuracy.

The experiment was controlled using Psychophysics Toolbox Version 3 [Brainard, 1997]. Data collection started with a standard nine-point calibration and validation. As explained below, participants viewed images blocked by pairs of classes; the images within a block were presented in random order (a new sequence was generated for each participant). Each trial started with a central fixation cross, displayed for 500 ms, after

which the image was displayed. The participant's task was to press one of the two response buttons to indicate the class to which the object(s) in the image belonged, after which the next trial was started automatically. Drift correction was performed after every 20 images; re-calibration was performed if the correction showed that this was necessary (after approximately every 200 images). Participants were offered a five minute break every 30 minutes.

The images in the Pascal dataset differ in size, and they are all smaller than the screen resolution used for the experiment (1,680 $\times$ 1,050 pixels). Instead of re-scaling the images to this resolution (which would result in visual artifacts and make the task unnatural), we presented the images in their original size, but at random offset from the center of the screen. This has the advantage that participants cannot easily develop a viewing strategy (e.g. always looking at the center of the screen, looking in the upper half), thus ensuring that we obtain maximally informative eye movement data.

3.3.4 Participants

A total of 28 participants (11 male) took part in the data collection, all students at the University of Edinburgh. They gave informed consent and were paid £10 per hour. The materials were divided into seven blocks of around 1,000 images each, where each block contained of all the images in one pair of classes, except for one large pairs of classes (cat/dog), which was split across three blocks. Each participant saw all the images in one block, except for five participants, who saw between two and four blocks in multiple sessions. Blocks were distributed across participants such that every image in every block was seen by five distinct participants.

3.3.5 Results

A total of around 178,000 fixations were collected, corresponding a mean of 5.7 fixations per participant per image. For example images with fixations superimposed, see Figure 3.2. Note that we removed the first fixation of each trial, as it almost always falls on the location of the fixation cross, and thus is uninformative (as well as often being outside the image, due to the off-center presentation).

The mean response time per image (the time from the onset of the image to the button press) was 889 ms, ranging from 786 ms for cat 1090 ms for cow. This indicates that the task can be performed very efficiently by human observers, especially compared to the 35 seconds reported in Su et al. [2012] as the time required to draw and

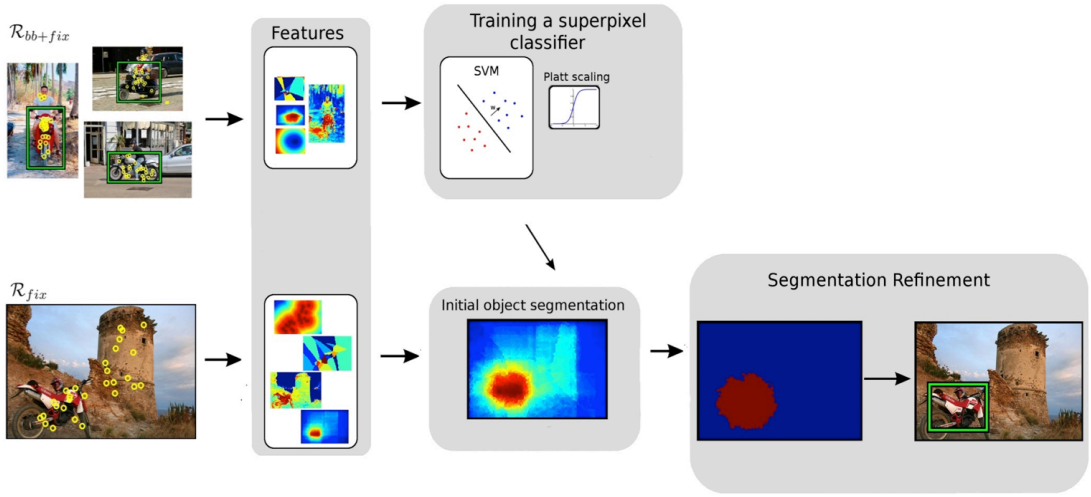


Figure 3.3: *Illustration of our method for predicting bounding-boxes from fixations. See Section 3.4.*

verify a tight bounding box. Participants were nevertheless highly accurate at the task, with a mean discrimination accuracy (percentage of correct button presses) of 95.2%, ranging from 92.9% for sofa/diningtable to 96.1% for boat/aeroplane.

We then compared the positions of the human fixations with the locations of the ground-truth bounding-boxes provided with the Pascal dataset. On average, 75.2% of all fixations on an image fell within one of the bounding-boxes for that image, ranging from 57.3% for boat to 89.6% for cat. This provides initial evidence of our claim that fixation data is useful for localizing objects, though it also indicates that there is considerable inter-class variability in fixation behavior.

3.4 From fixations to bounding-boxes

We present here our method for localizing objects in an image given fixations (Figure 3.1). We model this problem as figure-ground segmentation. Our model takes as input human right eye and left eye fixations Φ for an image I in $\mathcal{R}_{fix}$ and infers the spatial support of the object by labeling each pixel as either object or background. The method has two stages (Figure 3.3):

I) **Initial object segmentation** (Section 3.4.1) The first stage predicts an initial estimate of the object position by labeling each (super-)pixel individually. This predictor captures the relation between fixations and object positions. It is trained on the image set $\mathcal{R}_{bb+fix}$, which is annotated with both fixations and manual

bounding-boxes. The output of this stage is a values for each pixel of I that corresponds to the probability to be on the object.

II) **Segmentation refinement** (Section 3.4.2). The second stage refines the segmentation with an energy model similar to GrabCut [Rother et al., 2004]. This includes pairwise dependencies between neighboring superpixels, acting as a prior preferring spatially smooth segmentations.

3.4.1 Initial object segmentation

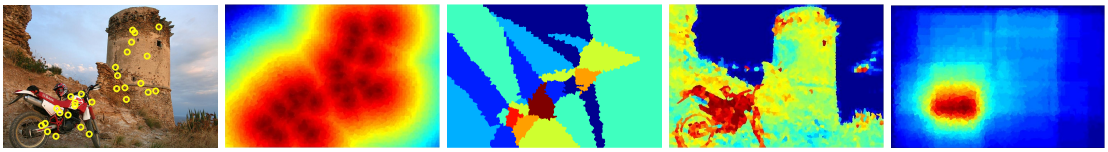
We describe here the first stage of deriving object segmentations from fixations. We use an SVM to classify superpixels into object or background based on a diverse set of features computed from the fixations, such as the distance between a superpixel and the nearest available fixation. Before the classifier is ready to label superpixels in a new image, we train it on the set $\mathcal{R}_{bb+fix}$. In this fashion it can learn a relation between fixations and object positions specific for that object class (i.e. the relation between these features and the fact that a superpixel is on the object or not). After training, the classifier is applied to each image $I \in \mathcal{R}_{fix}$, resulting in a soft segmentation mask M (Figure 3.3). Each pixel value in M corresponds to the estimated probability for it to be on the object.

Features. We start by over-segmenting each image I into superpixels S using the Turbopixels method [Levinshtein et al., 2009]. Operating at the superpixel level greatly reduces the computational cost and memory requirements of our technique.

Let Φ be the set of fixations in the image. Each fixation is determined by four values: the (x,y) position, the duration and the rank of the fixation in chronological order.

Fixation position features. As mentioned in Section 3.3, we acquired fixations under a visual search task instead of the usual free-viewing task to increase the proportion of fixations on (or near) the target object. Therefore, the position of the fixations directly relates to the position of the object. This motivates us to construct good features indicating whether a superpixel s lies inside the object based on its position relative to the fixations (Figure 3.5a):

- *mean distance:* the distance between the center of s and the mean position of all fixations in the image.



(a) (b) (c) (d) (e)

Figure 3.4: *(a) The positions of the fixations in the image. (b) The* near distance *feature. (c) The* rank *feature. (d) The* appearance *feature according to the distance of the nearest fixation. (e) The* objectness *feature.*

- *near distance:* the distance between the center of s and the position of the fixation nearest to s (Figure 3.4b).
- *mean offsets:* the vertical and horizontal difference between the center of s and the mean position of all fixations in the image.
- *near offsets:* the vertical and horizontal difference between the center of s and the position of the fixation nearest to s.

Fixation timing features. In addition to the position of each fixation, the eye tracker also delivers timing information, such as the duration and rank of each fixation. These properties carry valuable information. Intuitively, the longer a fixation lasts, the more significant it is. Moreover, in most images the first few fixations will not fall on the target object yet, as the annotator is searching for it, while later fixations are likely to be on the target (Figure 3.4c).

- *duration:* the duration of the fixation nearest to s.
- *rank:* the rank of the fixation nearest to s.

Fixation appearance features. The features considered so far support learning a *fixed* spatial relation between the fixations and the superpixels on the object. For example, we could learn that superpixels within a certain distance of the nearest fixation are likely to be on the object. Or we could learn that most of the object mass is below the mean fixation position (e.g. for the person class, as faces receive most fixations).

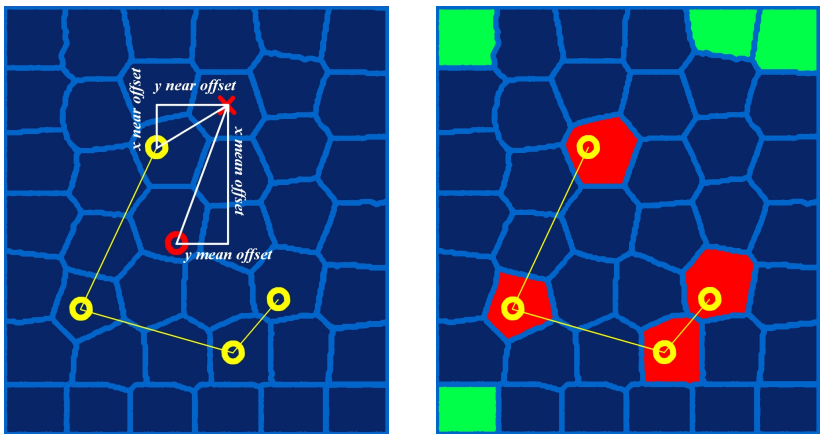
However, in images of natural objects this spatial relation might change from image to image and therefore might be hard to recover when reasoning about image coordinates alone. For instance, animals can appear in a wide range of viewpoint and

deformations, but viewers tend to fixate mostly their heads. This makes it difficult to guess the full extent of their body based purely on the fixation positions.

Here we consider another family of features based on the *appearance* of superpixels (i.e. their color distribution). The RGB color of a superpixel is computed as the average color of the constituent pixels. The key idea is that, while the fixations do not cover the whole object, many of them are on the object, and so provide examples of how it looks. Hence, we can learn about object appearance from a few superpixels hit by fixations. Analogously, we can learn about background appearance from superpixels far from fixations. We can then transfer this knowledge to all superpixels in the image. For example we might find out that a horse is brown and the background is green, even from just a few fixations, and use this knowledge to segment out the whole horse (Figure 3.1). Note that this idea works regardless of the spatial relation between the shape and size of the horse in the image and the fixation positions. It effectively creates a mapping from fixation to segmentations that *adapts* to the contents of the target image.

More precisely, we estimate two Gaussian Mixture Models (GMM), one for the object and one for the background. Each GMM has 5 components, each of which is a full-covariance Gaussian over the RGB color space. We estimate the object GMM A_{obj} from all pixels inside all superpixels hit by any fixation, as these are likely on the object (Figure 3.5b). Selecting background sample pixels is harder as the converse relation does not hold: the fact that a superpixel is not hit by any fixation does not reveal much about it being on the background or the object (in fact nearly all superpixels are not hit by a fixation, as fixations are extremely sparse). Therefore, we sample superpixels according to three criteria, leading to three different background GMM models A_{bg}, and we leave it to the learner to decide how to best weight them: (1) sample proportionally to the distance to the mean fixation, resulting in many samples far from the mean fixation and less and less samples as we get nearer; (2) sample proportionally to the distance to the nearest fixation. This is simply an alternative way to express 'far from the expected object location'; (3) sample inversely proportionally to the objectness probability (see next paragraph).

After estimating the appearance models A_{obj}, A_{bg}, we use them to evaluate every superpixel s in the image, resulting in per-pixel likelihoods of object/background $p(s|obj) = A_{obj}(s), p(s|bg) = A_{bg}(s)$. We combine these likelihoods in a per-pixel posterior probability of being on the object with Bayes' formula under a uniform prior: $p(obj|s) = p(s|obj)/(p(s|obj) + p(s|bg))$ (Figure 3.4d). We compute three different



(a) (b)

Figure 3.5: *Feature extraction.* (a) *Fixation position features. The yellow circles indicate the positions of the fixations in the image, while the red circle indicates the mean position of all fixations. The figure shows the distances from a superpixel center (red cross) that are required to compute all the feature position features.* (b) *Selecting foreground and background superpixels to estimate appearance models for the fixation appearance features. The selected foreground superpixels are indicated in red, while the sampled background superpixels according to the distance from the nearest fixation are in green.*

posteriors, by using each of the three background models in turn, resulting in three appearance features for each super-pixel.

Objectness feature. As an additional feature, we incorporate the *objectness* measure [Alexe et al., 2010]. Objectness estimates the probability that an image window contains an object of *any* class. It is designed to distinguish windows containing an object with a well defined boundary and center, such as cows and telephones, from amorphous background windows, such as grass and road. It measures distinctive characteristics of objects, such as appearing different from their surroundings, having a closed boundary, and sometimes being unique within the image.

We sample 1,000 windows using Alexe et al. [2010] according to their probability of containing an object. Then we convert this measure to per-superpixel probabilities by averaging the objectness value over all windows covering a super-pixel. This objectness feature helps by pushing the segmentation method towards objects and away from backgrounds (Figure 3.4e).

Training a superpixel classifier. We train a separate classifier for each object class, as the relation between fixations and objects can be complex and vary between classes. As

training samples we use the feature vectors of all superpixels from the $\mathcal{R}_{bb+fix}$ images of a class (Figure 3.3). Each superpixel is labeled according to whether it is inside a ground-truth bounding-box or not. After whitening the features, we train a linear SVM with the very efficient implementation of Vedaldi and Fulkerson [2008] on a random 80% subset of the training data. We set the regularization parameter C by validation on a held-out 10% of the data, and then re-train the SVM on the total 90% of the data. In order to get a smooth, probabilistic output we apply Platt scaling [Platt, 1999] and fit a sigmoid to the output of the SVM on the remaining 10% of the training data.

Applying the classifier to a new image $I \in \mathcal{R}_{fix}$ yields a soft-segmentation mask M where each pixel value corresponds to the estimated probability of being on the target object. This is the output of the first stage of our method.

3.4.2 Segmentation refinement

We describe here the second stage of deriving object segmentations from fixations. This refines the soft-segmentation M output by the first stage by taking into account pairwise dependencies between neighboring superpixels and by improving the appearance models.

Let $l_s \in \{0,1\}$ be the label for superpixel s and L be the labeling of all l_s in the image. We employ a binary pairwise energy function E defined over the superpixels and their labels, analog to Kuettel and Ferrari [2012]; Rother et al. [2004]:

$$E(L) = \sum_s M_s(l_s) + \sum_s A_s(l_s) + \sum_{s,r} V(l_s, l_r) \tag{3.1}$$

As in Kuettel and Ferrari [2012]; Rother et al. [2004], the pairwise potential V encourages smoothness by penalizing neighboring pixels taking different labels. The penalty depends on the color contrast between the pixels, being smaller in regions of high contrast (image edges). The summation over (s,r) is defined on an eight-connected pixel grid.

The linear combination of the two unary potentials A_s and M_s evaluate how likely a superpixel s is to take label l_s according to an appearance model and a location model.

Because of the probabilistic nature of the soft segmentation mask M, we can use $M_s(l_s) = M(s)^{l_s}(1-M(s))^{1-l_s}$ as a unary potential (with M_s the value of the mask at superpixel s). As M_s estimates the probability that superpixel s is on the object, this potential encourages the final segmentation to be close to M (see Kuettel and Ferrari

[2012]). This anchors the segmentation to image regions likely to contain the target object, while letting this second stage refine its exact delineation.

The second unary potential A_s evaluates how likely a superpixel s is to take label l_s, according to object and background appearance models. As in the classic GrabCut [Rother et al., 2004], an appearance model consists of two GMMs, one for the object (used when $l_s = 1$) and one for the background (used when $l_s = 0$). Each GMM has five components, each a full-covariance Gaussian over the RGB color space.

In traditional work using similar energy models [Blake et al., 2004; Rother et al., 2004; Wang and Cohen, 2005], estimating the appearance models requires user interaction to indicate the image region containing the object (typically a manually drawn bounding-box or scribbles). Recently, Kuettel and Ferrari [2012] proposed to automatically estimate the appearance models from a soft segmentation mask produced by transferring segmentations from manually annotated images in a training set. Inspired by that idea, we estimate our appearance models from the mask M obtained from fixations in Section 3.4.1 (so our method does not require training segmentations).

After this initial estimation, we follow Rother et al. [2004] and alternate between finding the optimal segmentation L given the appearance models, and updating the appearance models given the segmentation. The first step is solved globally optimally by minimizing (3.1) using graph-cuts [Boykov and Kolmogorov, 2004] as our pairwise potentials are submodular. The second step fits GMMs to labeled superpixels. Defining the energy over superpixels [Guillaumin et al., 2014; Ladicky et al., 2009; Veksler et al., 2010] instead of pixels brings great memory savings and reduces the cost to optimize over L. As demonstrated in Guillaumin et al. [2014], the accuracy of the superpixel model is essentially identical to the corresponding pixel model.

The final output of our method is a bounding-box enclosing the largest connected component in the segmentation (Figure 3.3).

3.5 Experiments

We carry out experiments on the challenging Pascal VOC 2012 benchmark [Everingham et al., 2012]. We first evaluate the ability of our method to derive object bounding-boxes from fixations on the trainval part of the dataset (Section 3.5.1). Next, we train the object class detector of Felzenszwalb et al. [2010] from these bounding-boxes and compare its performance on the test set to the original model trained from ground-truth bounding-boxes (Section 3.5.2).

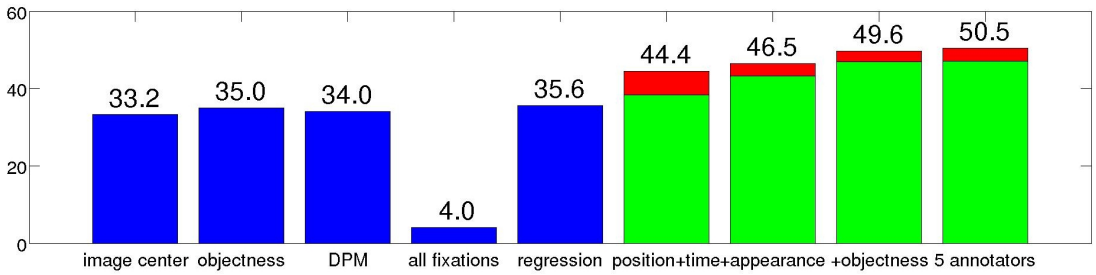


Figure 3.6: *CorLoc performance for the baselines (blue bars) and several stripped down versions of our model (green+red) bars. The height of the green bars indicate the performance of the initial segmentation method (i) using only position and timing features, (ii) adding appearance features, (iii) adding the objectness feature, and (iv) using all features and fixations from 5 annotators (the previous three bars use only 2 annotators). The height of the red bars show the improvement brought by the refinement stage.*

3.5.1 From fixations to bounding-boxes

Data. We consider 10 classes from Pascal VOC 2012, for a total of 6,270 images in the trainval part (see Section 3.3.1 for a list of classes). We split each class into two subsets $\mathcal{R}_{bb+fix}$ and $\mathcal{R}_{fix}$. We use both ground-truth bounding-boxes and human fixations from the set $\mathcal{R}_{bb+fix}$ to train our segmentation model. This subset contains only 7% of all trainval images. This fraction was chosen so that each class has ≥ 20 images. On average, each class has only 44 images. After training, we use our model to predict bounding-boxes for images in the $\mathcal{R}_{fix}$ set, which are annotated only by fixations.

Evaluation measure. We report performance as CorLoc [Deselaers et al., 2010], i.e. the percentage of images in $\mathcal{R}_{fix}$ where a method correctly localizes an object of the target class according to the PASCAL criterion [Everingham et al., 2010] (intersection-over-union ≥ 0.5). This measure is commonly used in WSOL [Deselaers et al., 2010; Pandey and Lazebnik, 2011; Siva and Xiang, 2011; Deselaers et al., 2012; Russakovsky et al., 2012; Siva et al., 2012; Shi et al., 2013; Bilen et al., 2014; Cinbis et al., 2014; Shi et al., 2015; Cinbis et al., 2016; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016].

As most previous WSOL methods [Deselaers et al., 2010; Pandey and Lazebnik, 2011; Siva and Xiang, 2011; Deselaers et al., 2012; Russakovsky et al., 2012; Siva et al., 2012; Shi et al., 2013; Bilen et al., 2014; Cinbis et al., 2014; Shi et al., 2015; Cinbis et al., 2016; Wang et al., 2015; Bilen and Vedaldi, 2016; Kantorov et al., 2016], our scheme returns exactly one bounding box per class per image in $\mathcal{R}_{fix}$. This enables

clean comparisons in terms of CorLoc on this set, and keeps the eye tracking task simple. If multiple instances of the same class appeared in an image, different users may fixate of different instances. It is worth noting that our model is able to detect one of them (i.e., by fitting a bounding box around the largest connecting component. Finding all connected components in the output mask and fitting a bounding box around each of them could enable us to evaluate the ability of our method to detect multiple objects per image.

Baselines. In Figure 3.6 we evaluate several informative baselines: **(1) image center:** a window in the image center with area set to the average over the object bounding-boxes in the $\mathcal{R}_{bb+fix}$ set. This baseline provides an indication of the difficulty of the dataset; **(2) all fixations:** a bounding-box around all fixations; **(3) objectness:** the window with the highest objectness probability, out of 1000 sampled using Alexe et al. [2010]; **(4) DPM:** the highest scored detection returned by a Deformable Parts Model detector Felzenszwalb et al. [2010] trained on the $\mathcal{R}_{bb+fix}$ set; **(5) regression:** linear regression from the mean and variance of the fixation positions to a bounding-box. This models the spatial relation between the cloud of fixations and the object with a single translation and anisotropic scaling transformation, constant across all images of a class.

As Figure 3.6 shows, the *image center* baseline achieves 33.2% CorLoc, confirming the dataset contains mostly images with smaller, often off-center objects. Interestingly, *all fixations* fails entirely, demonstrating that the task of deriving bounding-boxes from fixations is far from trivial. *Regression* does much better, finding the object in 35.6% of the images. This highlights the need for learning the relation between fixations and bounding-boxes. Note how *Objectness* also performs quite well (35.0%). It can find some objects even when used on its own, confirming observations made by earlier research [Guillaumin and Ferrari, 2012]. As *regression* and *objectness* incorporate elements of our full method, they set the standards to beat. Finally, the *DPM* baseline reaches only 34.0% CorLoc, showing that the problem cannot be solved simply by training an object detector on the small fully annotated set $\mathcal{R}_{bb+fix}$.

Results. Figure 3.6 shows the CorLoc achieved by several stripped down version of our model. We study the impact of using increasingly more features (Section 3.4.1), and the difference between using only the initial segmentation stage (Section 3.4.1) or also the segmentation refinement stage (Section 3.4.2). To quantify the performance of the initial segmentation stage, we threshold the soft-segmentation mask M and fit a

bounding-box around the largest segment. The threshold is optimized on the training set $\mathcal{R}_{bb+fix}$.

The results reveal several interesting observations: (1) all feature types we propose contribute to the overall performance of the full model, as adding each type in turn progressively leads to higher CorLoc; (2) with 49.6% CorLoc, our full model significantly outperforms all baselines, including *regression* and *objectness*. This shows that it can learn better, more complex relations between the fixations and the object's spatial extent than the regression baseline. It also shows that, while objectness is a valuable cue to the position of objects, our model goes well beyond it; (3) the segmentation refinement stage always helps, adding between 3% and 6% CorLoc depending on the features used in the initial segmentation stage.

As a reference, the weakly supervised method of Siva et al. [2013], which produces bounding-boxes on a set of images labeled only as containing a class, achieves 32% in a similar setting on PASCAL VOC 2007 (which is a dataset of similar difficulty). This method was the state of the WSOL at the submission time of our ECCV 2014 paper [Papadopoulos et al., 2014], where we published the research covered in this chapter. The current state-of-the-art WSOL method of Bilen and Vedaldi [2016] on PASCAL VOC 2007 achieves 54.2% CorLoc with a single CNN model. It is important to note that our work provides a different way to reduce annotation effort which is complementary to other WSOL techniques (Section 2.2). It could potentially be integrated with some of them for even greater savings. To demonstrate this point, in Section 5.6 we present a method that incorporates eye tracking data into a standard WSOL framework and achieves a significant improvement over WSOL with a modest extra annotation cost.

Our results discussed above used fixations from just *two* annotators. This is an especially economical annotation scenario, as it takes only a total of about *two seconds* to annotate an object, substantially less than the 35 seconds it takes to draw a bounding-box Su et al. [2012]. However, as we collected fixations from five annotators per image (Section 3.3), for completeness we report also the performance of our model when using all annotators (rightmost bar in Figure 3.6). As performance improves only marginally, we conclude that our model efficiently exploits the information provided by two annotators, and keep this as the default setting in the next experiment. At this point we should note that in our model we simply use all the right and left eye fixations of each annotator independently as the set of fixations Φ and we do not average or discard any of them.

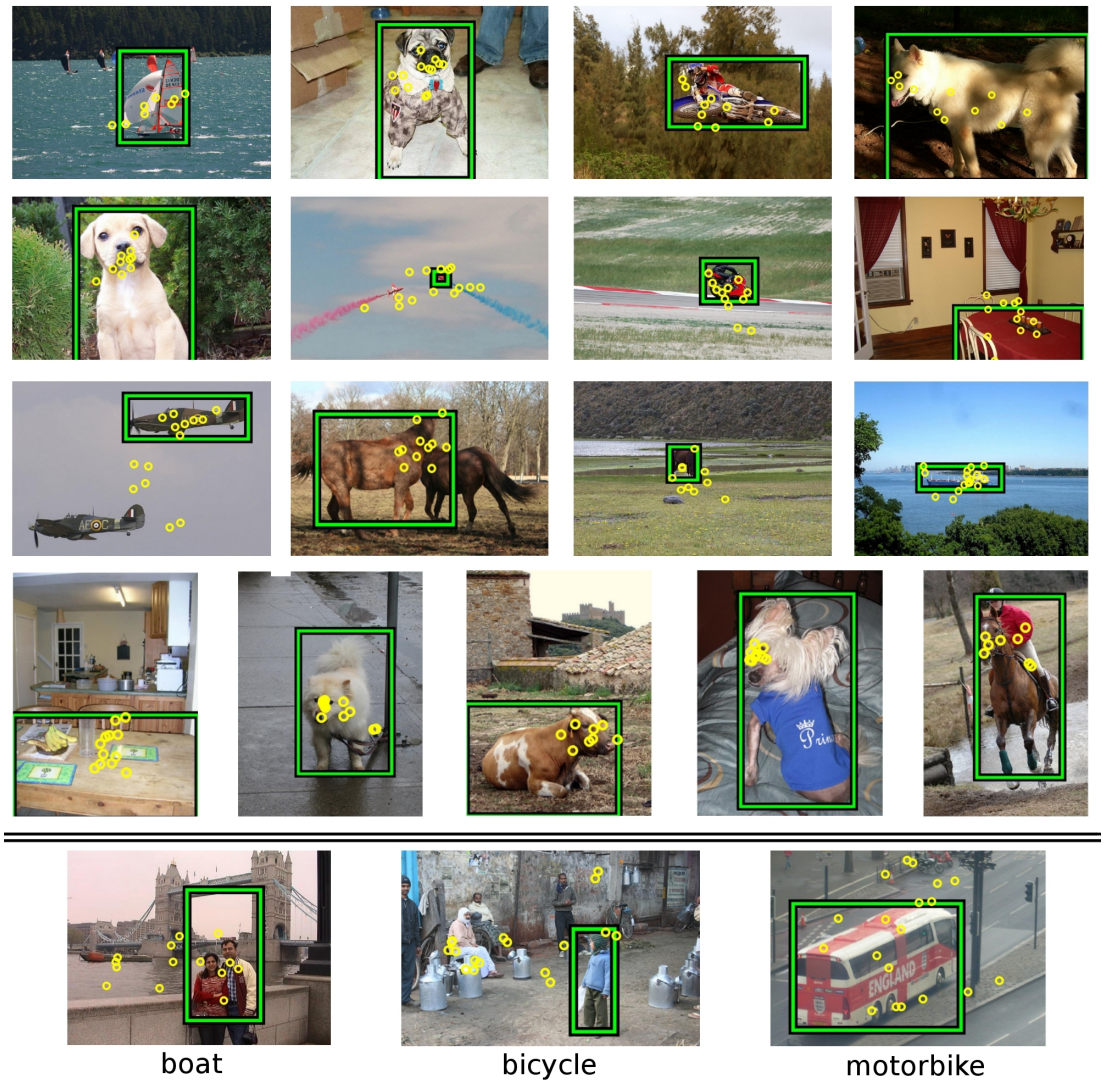


Figure 3.7: *Qualitative results of our method for localizing objects given fixations. The fixations of two different annotators are indicated with yellow circles while the predicted bounding-boxes are in green. Successful examples are shown in the first four rows for various classes, while some failure cases are shown in the last row. Note how our method nicely outputs a bounding-box covering the whole object even when the fixations are concentrated on only part of it, often the center, or the head.*

3.5.2 Training object class detectors from fixations

Settings. In the previous subsection we automatically derived bounding-boxes from fixations for the $\mathcal{R}_{fix}$ part of the Pascal VOC 2012 trainval set (i.e. 93% of the total). Here we use these predicted bounding-boxes, along with the 7% images in $\mathcal{R}_{bb+fix}$ with ground-truth bounding-boxes, to train a DPM detector [Felzenszwalb et al., 2010]. Fol-

	All ground-truth bounding boxes	All predicted bounding boxes	7.8× fewer ground-truth boxes
DPM	25.5	12.5	13.1

Table 3.1: *The mAP performance of DPM detectors on the test set of PASCAL VOC 2012 dataset using different types of box annotations.*

lowing the standard Pascal VOC protocol, the negative training set for a class contains all trainval images not containing that class. After training, the detectors are applied to the Pascal VOC 2012 test set (10,991 images). Performance is quantified by the mean average precision (mAP) over all 10 classes, as returned by the Pascal VOC evaluation server. We compare performance to detectors trained from exactly the same images, but *all* annotated with ground-truth bounding-boxes.

Results. The mAP of DPM detectors trained from bounding boxes derived by our method is 12.5%, compared to 25.5% for the detectors trained from ground-truth bounding boxes. We consider this an encouraging result, given that our scenario enables to train these detectors in 7.8× less total annotation time compared to drawing bounding boxes on all images. This estimate takes into account all relevant factors, i.e. two annotators per image at one second per image, the extra time of displaying the fixations cross preceding every image (500ms), the time to set up and calibrate the eye tracker, breaks between blocks of images, and the time to draw bounding-boxes on the 7% images in $\mathcal{R}_{bb+fix}$. Interestingly, training DPMs from ground-truth bounding-boxes for a 7.8× smaller training set leads to comparable performance as our method (13.1%).

3.6 Conclusions

We have presented a novel approach to train object detectors. Instead of the traditional, time consuming manual bounding-box annotation protocol, we proposed to learn the detector from eye movement data recorded while annotators simply look for the object in the training images. We proposed a technique to successfully derive object bounding boxes from such eye movement data and demonstrated that they can be used to train an object class detector [Felzenszwalb et al., 2010]. In its current form, when given equal total annotation time, our scheme leads to detectors that are about as good as those trained from manual bounding-box annotations.

We found out that our visual search task can be performed very efficiently (0.9 seconds per image per annotator) and in fact this time 19× faster than the time required to draw a bounding box [Su et al., 2012] if we consider using fixations from two annotators. However, our scheme has some disadvantages that increase the total annotation time. Most of the annotation time of our scheme (43%) is spent on manually annotating the bounding boxes on the small $\mathcal{R}_{bb+fix}$ set. As mentioned in this chapter, we use this set to learn the relation between fixations and bounding boxes. Note that the actual time of the visual search task is only the 30% of the total annotation time of our scheme. The rest 27% of the time is spend on setting up the experiment, calibrating the eye tracker and looking at the fixation cross preceding every image (500ms).

Another limitation of this scheme is that currently it is quite difficult to crowdsource eye tracking data, which is essential for building a very large scale dataset. Very cheap eye trackers are right now available but the task requires the annotators to have access to this special equipment. Being able to track accurately eye movements with conventional web cameras could bypass this problem and make crowd-sourcing viable for eye tracking data [Skovsgaard et al., 2013; Xiao et al., 2015; Xu et al., 2015; Wood et al., 2015; Krafka et al., 2016; Papoutsaki et al., 2016]

For all these reasons, in the following chapters of this thesis, we introduce ideas leading to annotation schemes that do not require any manually drawn bounding boxes or any special equipment (e.g., eye tracker). This leads to even greater savings in annotation time and more importantly, our schemes presented in Chapters 4, 5 lead to detectors that perform in a range close to those trained from manually drawn bounding boxes.

Chapter 4

Training object class detectors using only human verification

Contents

4.1 Introduction

In the previous chapter we proposed a scheme for learning object detectors with eye tracking data. Our scheme was very efficient and we demonstrated that we can successfully derive object bounding boxes from eye movement data. However, as discussed in Section 3.6, it has some disadvantages: it requires drawing manually bounding box in a small set of images and it demands special equipment (eye tracker) that makes it unfeasible to crowd-source these kind of annotation data.

In this chapter, we propose another efficient annotation scheme which only requires humans to *verify* bounding boxes produced automatically by the learning algorithm: the annotator merely needs to decide whether a bounding box is correct or not. Crucially, answering this verification question takes much less time than actually drawing the bounding box. More importantly, this scheme overcomes the disadvantages of the eye tracking scheme as it eliminates the need to draw *any* bounding box and these human verifications are easily crowdsourceable.

Given a set of training images with image-level labels, our scheme iteratively alternates between updating object detectors, re-localizing objects in the training images, and querying humans for verification. At each iteration we use the verification signal in two ways. First, we update the object class detector using only positively verified bounding boxes. This makes it stronger than when using all detected bounding boxes, as it is commonly done in the weakly supervised setting, because typically many of them are incorrect. Moreover, once the object location in an image has been positively verified, it can be fixed and removed from consideration in subsequent iterations. Second, we observe that bounding boxes judged as incorrect still provide valuable information about where the object is not. Building on this observation, we use the negatively verified bounding boxes to reduce the search space of possible object locations in subsequent iterations. Both these points help to rapidly find more objects in the remaining images. This results in a framework for training object detectors which minimizes human annotation effort and eliminates the need to draw *any* bounding box.

In extensive experiments on the popular PASCAL VOC 2007 dataset [Everingham

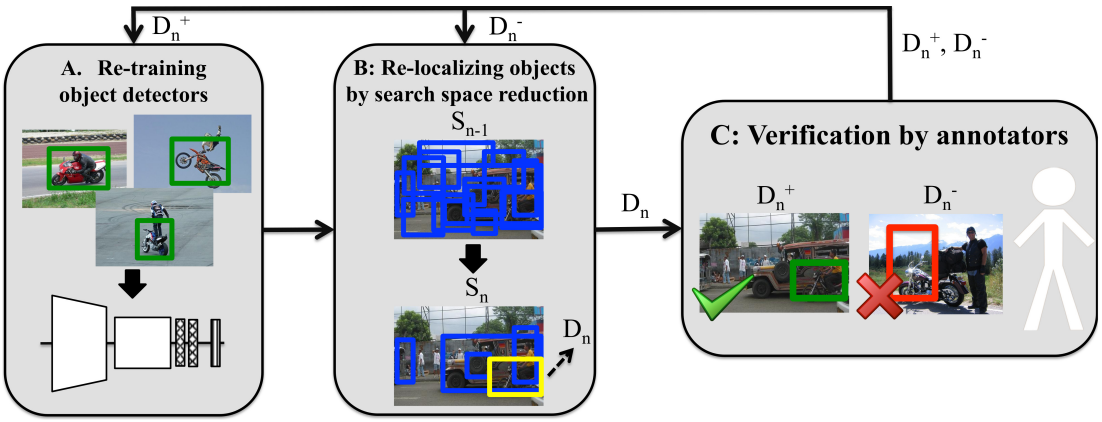


Figure 4.1: *Our framework iterates between (A) re-training object detectors, (B) re-localizing objects, and (C) querying annotators for verification. The verification signal resulting from (C) is used in both (A) and (B).*

et al., 2010] with both simulated annotators, expect annotators and annotators in Amazon Mechanical Turk (AMT), we show that: (1) using human verification to update detectors and reduce the search space leads to rapid production of high-quality bounding box annotations; (2) our scheme delivers object class detectors performing almost as well as those trained in a fully supervised setting, *without ever drawing any bounding box*; (3) as the verification task is very quick, our scheme substantially reduces total annotation time by 8×.

4.2 Related Work

The work of this Chapter is related to work reducing annotation time for training object detectors. More details about WSOL or other ways to reduce annotation effort for training object class detectors can be found in Section 2.2, 2.3. We focus here on human in the loop and active learning approaches that were not covered in Chapter 2.

4.2.1 Humans in the loop

Human-machine collaboration approaches have been successfully used in tasks that are currently too difficult to be solved by computer vision alone, such as fine-grained visual recognition [Branson et al., 2010; Deng et al., 2013; Wah et al., 2011, 2014], semi-supervised clustering [Lad and Parikh, 2014], attribute-based image classification [Biswas and Parikh, 2013; Parikh and Grauman, 2011; Parkash and Parikh, 2012]. These approaches combine the responses of pre-trained computer vision models on

a new test image with human input to fully solve the task. In the domain of object detection, Russakovsky et al. [2015b] propose such a scheme to fully detect all objects in images of complex scenes. Importantly, their object detectors are pre-trained on bounding-boxes from the large training set of ILSVRC 2014 [Russakovsky et al., 2015a], as their goal is not to make an efficient training scheme.

4.2.2 Active learning

Active learning schemes iteratively train models while requesting humans to annotate a subset of the data points actively selected by the learner as being the most informative. Previous active learning work has mainly focused on image classification [Joshi et al., 2009; Kapoor et al., 2007; Kovashka et al., 2011; Qi et al., 2008], and free-form region labeling [Siddiquie and Gupta, 2010; Vijayanarasimhan and Grauman, 2008, 2009].

A few researchers have proposed active learning schemes specifically for training object class detectors [Vijayanarasimhan and Grauman, 2014; Yao et al., 2012]. Vijayanarasimhan and Grauman [2014] propose an approach where the training images do not come from a predefined dataset but are crawled from the web. Here annotators are asked to draw many bounding-boxes around the target objects (about one third of the training images [Vijayanarasimhan and Grauman, 2014]). Yao et al. [2012] propose to manually correct bounding-boxes detected in video. While both Vijayanarasimhan and Grauman [2014]; Yao et al. [2012] produce high quality detectors, they achieve only moderate gains in annotation time, because drawing or correcting bounding-boxes is expensive. In contrast, our scheme only asks annotators to verify bounding-boxes, never to draw. This leads to more substantial reductions in annotation time.

4.3 Method

In this chapter we are given a training set with image-level labels. Our goal is to obtain object instances annotated by bounding-boxes and to train good object detectors while minimizing human annotation effort. We therefore propose a framework where annotators only need to *verify* bounding-boxes automatically produced by our scheme.

Our framework iteratively alternates between (A) re-training object detectors, (B) re-localizing objects in the training images, and (C) querying annotators for verification (Figure 4.1). Importantly, we use verification signals to help both re-training and

re-localization.

More formally, let I_n be the set of images for which we do not have positively verified bounding-boxes at iteration n yet. Let S_n be the corresponding set of possible object locations. Initially, I_0 is the complete training set and S_0 is a complete set of object proposals [Alexe et al., 2010; Zitnick and Dollár, 2014; Uijlings et al., 2013] extracted from these images (we use EdgeBoxes [Zitnick and Dollár, 2014]). To facilitate exposition, we describe our framework starting from the verification step (C, Section 4.3.1). At iteration n we have a set of automatically detected bounding-boxes D_n which are given to annotators to be verified. Detections which are judged to be correct $D_n^+ \subseteq D_n$ are used for re-training the object detectors (A) in the next iteration (Section 4.3.2). The verification signal is also used to reduce the search space S_{n+1} for re-localization (B, Section. 4.3.3). We describe our main three steps below. We defer to Section 4.4 a description of the object detection model we use, and of how to automatically obtain initial detections D_0 to start the process.

4.3.1 Verification by annotators

In this phase, we ask annotators to verify the automatically generated detections D_n at iteration n. For this we explore two strategies (Figure 4.2): simple yes/no verification, and more elaborate verification in which annotators are asked to categorize the type of error.

Yes/No Verification. In this task the annotators are shown a detection l_d and a class label. They are instructed to respond Yes if the detection correctly localizes an object of that class, and No otherwise. This splits the set of object detections D_n into D_n^+ and D_n^-. We define "correct localization" based on the standard PASCAL Intersection-over-Union criterion [Everingham et al., 2010] (IoU). Let l_d be the detected object bounding-box and l_{gt} be the actual object bounding-box (which is not given to the annotator). Let $\text{IoU}(l_a, l_b) = |l_a \cap l_b| / |l_a \cup l_b|$, where $|\cdot|$ denotes area. If $\text{IoU}(l_{gt}, l_d) \geq 0.5$, the detected bounding-box should be considered correct and the annotator should answer Yes. Intuitively, this Yes/No verification is a relatively simple task which should translate into fast annotation times.

Yes/Part/Container/Mixed/Missed Verification. In this task, we asked the annotators to label an object detection l_d as Yes (correct), Part, Container, Mixed, or Missed. Yes is defined as above ($\text{IoU}(l_{gt}, l_d) \geq 0.5$). For incorrect detections the annotators

Figure 4.2: *Our two verification strategies for some images of the dog class. Yes/No verification (left): verify a detection as either correct (Yes) or incorrect (No). YPCMM verification (right): label a detection as Yes, Part, Container, Mixed or Missed.*

are asked to diagnose the error as either *Part* if it contains part of the target object and no background; *Container* if it contains the whole object and some background; *Mixed* if it contains part of the object and some background; *Missed* if the object was completely missed. This verification step splits D_n into D_n^+ and D_n^{ypcmm-}. Intuitively, determining the type of error is more difficult leading to longer annotation times, but also brings more information that we can use in the next steps.

4.3.2 Re-training object detectors

In this step we re-train object detectors. After the verification step we know that D_n^+ contains well localized object instances, while D_n^- or D_n^{ypcmm-} do not. Hence we train using only bounding-boxes $D_1^+ \cup \cdots \cup D_n^+$ that have been positively verified in past iterations. To obtain background training samples, we sample proposals which have an IoU in range $[0-0.5)$ with positively verified bounding boxes.

We should point out that this assumption for background samples has a risk of considering positive examples of other non-verified instances as background (false positives). We found experimentally that in PASCAL VOC and MS COCO, the number of those false positives is negligible and in practice it does not affect the training. We also considered sampling background training samples from proposals with an IoU in range $[0.1-0.5)$ with positively verified bounding boxes. Theoretically, this further reduces the risk for false positives because different instances are usually not highly overlapped in the images. However, in practice this does not affect the training of the detectors.

Note how it is common in WSOL [Bilen et al., 2014, 2015; Cinbis et al., 2014; Deselaers et al., 2010; Russakovsky et al., 2012; Siva and Xiang, 2011; Song et al., 2014a,b; Wang et al., 2015] to also have a re-training step. However, they typically use all detected bounding-boxes D_n. Since in WSOL generally less than half of them are correct, this leads to rather weak detectors. In contrast, our verification step enables us to train purely from correct bounding-boxes, resulting in stronger, more reliable object detectors.

4.3.3 Re-localizing objects by search space reduction

In this step we re-localize objects in the training images. For each image, we apply the current object detector to score the object proposals in it, and select the proposal with the highest score as the new detection for that image. Importantly, we do not evaluate *all* proposals S_0, but instead use the verification signal to reduce the search space by removing proposals.

Positively verified detections D_n^+ are correct by definition and therefore their images need not be considered in subsequent iterations, neither in the re-localization step nor in the verification step. For negatively verified detections we reduce the search space depending on the verification strategy, as described below.

Yes/No Verification. In the case where the annotator judges a detection as incorrect (D_n^-), we can simply eliminate its proposal from the search space. This results in the updated search space S_{n+1}, where one proposal has been removed from each image with an incorrect detection.

However, we might make a better use of the negative verification signal. Since an incorrect detection has an IoU < 0.5 with the true bounding-box, we can eliminate all proposals with an IoU ≥ 0.5 with it. This is a more aggressive reduction of the search space. While it may remove some proposals which are correct according to the IoU criterion, it will not remove the best possible proposal. Importantly, this strategy eliminates those areas of the search space that matter: high scoring locations which are unlikely to contain the object. In Section 4.6.2 we investigate which way of using negatively verified detection performs better in practice.

Yes/Part/Container/Mixed/Missed Verification. In the case where annotators categorize incorrect detections as Part/Container/Mixed/Missed, we can use the type of error to get an even greater reduction of the search space. Depending on the type of

error we eliminate different proposals (Figure 4.3): *Part:* eliminate all proposals which do not contain the detection; *Container:* eliminate all proposals which are not inside the detection; *Mixed:* eliminate all proposals which are not inside the detection, or do not contain it, or have zero IoU with it, or have IoU ≥ 0.5 with it; *Missed:* eliminate all proposals which have non-zero IoU with the detection.

To precisely determine what is "inside" and "contained", we introduce the Intersection-over-A measure: $\text{IoA}(l_a, l_b) = |l_a \cap l_b| / |l_a|$. Note that $\text{IoA}(l_{gt}, l_d) = 1$ if the detection l_d contains the true object bounding-box l_{gt}, whereas $\text{IoA}(l_d, l_{gt}) = 1$ if l_d covers a part of l_{gt}. In practice, if l_d is judged to be a *Part* by the annotator, we eliminate all proposals l_s with $\text{IoA}(l_d, l_s) \leq 0.9$. Similarly, if l_d is judged to be a *Container*, we eliminate all proposals l_s with $\text{IoA}(l_s, l_d) \leq 0.9$. We keep the tolerance threshold 0.9 fixed in all experiments.

Note how in WSOL there is also a re-localization step. However, because there is no verification signal there is also no search space reduction: each iteration needs to consider the complete set S_0 of proposals. In contrast, in our work the search space reduction greatly facilitates re-localization.

4.4 Implementation details

We summarize here two existing state-of-the-art components that we use in our framework: the object detection model, and a WSOL algorithm which we use to obtain initial object detections D_0.

4.4.1 Object class detector

As object detector we use Fast R-CNN [Girshick, 2015], which combines object proposals [Alexe et al., 2010; Zitnick and Dollár, 2014; Uijlings et al., 2013] with CNNs [He et al., 2014; Krizhevsky et al., 2012; Simonyan and Zisserman, 2015]. Instead of Selective Search [Uijlings et al., 2013] we use EdgeBoxes [Zitnick and Dollár, 2014] which gives us an "objectness" measure [Alexe et al., 2010] which we use in the initialization phase described below. For simplicity of implementation, for the re-training step (Section 4.3.2) we omit bounding-box regression, so that the set of object proposals stays fixed throughout all iterations. For evaluation on the test set, we then train detectors with bounding-box regression. In most of our experiments, we use AlexNet [Krizhevsky et al., 2012] as the underlying CNN architecture.

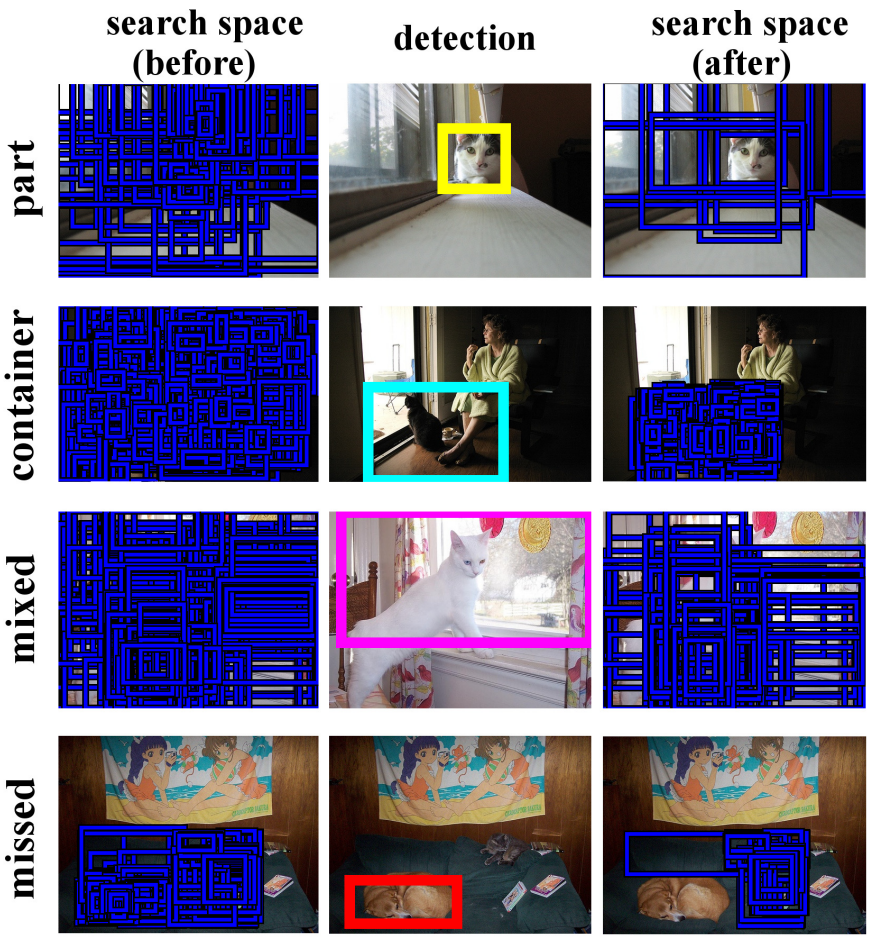


Figure 4.3: *Visualization of search space reduction induced by YPCMM verification on some images of the cat class (part, container, mixed, and missed). In the last row, the search space reduction steers the re-localization process towards the small cat on the right of the image and away from the dog on the left.*

4.4.2 Initialization by Multiple Instance Learning

We perform multiple instance learning (MIL) for weakly supervised object localization [Bilen et al., 2014; Cinbis et al., 2014; Song et al., 2014a] to obtain the initial set of detections D_0. We start with the training images I_0 and the set of object proposals S_0 extracted using EdgeBoxes [Zitnick and Dollár, 2014]. Following [Bilen et al., 2014; Girshick et al., 2014; Song et al., 2014a,b; Wang et al., 2015] we extract CNN features on top of which we train an SVM. We iterate between (A) re-training object detectors and (B) re-localizing objects in the training images. We stop when two subsequent re-localization steps yield the same detections, which typically happens within 10 iterations. These detections become D_0. In the very first iteration, we train the classifier using complete images as positive training examples [Cinbis et al., 2014; Russakovsky

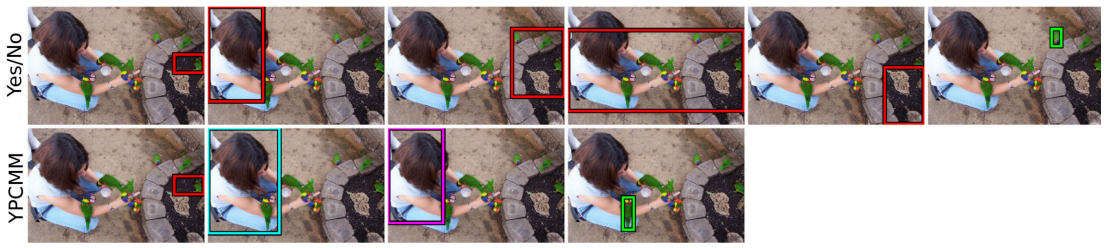


Figure 4.4: *Comparing the search process of Yes/No verification with Yes/Part/Container/Mixed/Missed on an image of the bird class. The extra signal for YPCMM allows for a more targeted search, resulting in fewer verification steps to find the object.*

et al., 2012]. More details about this standard MIL framework can be found in Section 2.2.

We apply two improvements to the standard MIL framework. First, in high dimensional feature space the discriminative SVM classifier can relatively easily separate any positive examples from negative examples, which means that most positive examples are far from the decision hyper-plane. Hence the same positive training examples used for re-training (A) are often re-localized in (B), leading to premature locked-in behavior. To prevent this, Cinbis et al. [2016] introduced multi-fold MIL: similar to cross-validation, the dataset is split into 10 subsets, where the re-localization on each subset is done using detectors trained on the union of all other subsets. Second, like in Cinbis et al. [2016]; Deselaers et al. [2010]; Guillaumin and Ferrari [2012]; Prest et al. [2012]; Shapovalova et al. [2012]; Siva and Xiang [2011]; Shi et al. [2012]; Tang et al. [2014]; Wang et al. [2014a], we combine the object detector score with a general measure of "objectness" [Alexe et al., 2010], which measures how likely it is that a proposal tightly encloses an object of any class (e.g. bird, car, sheep), as opposed to background (e.g. sky, water, grass). More particularly, we linearly combine these two scores under the assumption of equal weights. In this paper we use the recent objectness measure of Zitnick and Dollár [2014].

4.5 Collecting human verifications

We now describe how we collect human verifications using expert annotators in an in-house experiment (Section 4.5.1) and how we crowd-source them using naive annotators on Amazon Mechanical Turk (AMT) (Section 4.5.2).

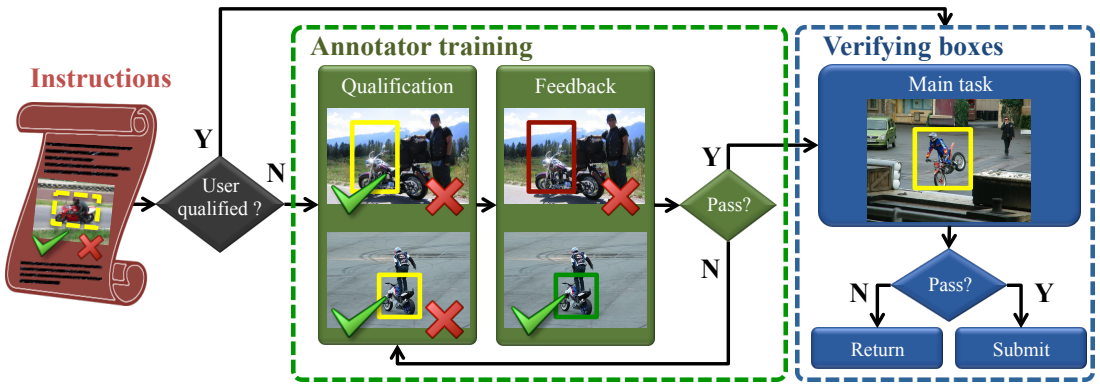


Figure 4.5: ***The workflow of our crowd-sourcing protocol for collecting Yes/No verification.*** *The annotators read a set of instructions and then go through an interactive training stage that consists of a qualification test at the end of which they receive a detailed feedback on how well they performed. Annotators who successfully pass the test can proceed to the annotation stage. In case of failure, they are allowed to repeat the test as many times as they want until they succeed.*

4.5.1 Expert human verification

For the Yes/No and YPCMM verification tasks, we used five annotators from the University of Edinburgh who were given examples to learn about the IoU criterion [Everingham et al., 2010]. For both tasks we created a full-screen interface. All images of a target class were shown in sequence, with the current detection superimposed. For Yes/No verification, annotators were asked to press "1" for Yes and "0" for No. For YPCMM verification, the annotators were asked to click on one of five on-screen buttons corresponding to Yes, Part, Container, Mixed and Missed.

4.5.2 Crowd-sourced human verifications

We now describe our crowd-sourcing framework for the Yes/No verification task (Figure 4.5). Annotators read a simple set of instructions and then go through an interactive training stage. Those who successfully pass the training stage can proceed to the main stage.

Instructions. The annotators are given an image with a displayed bounding box and the name of a target object class. They are instructed to press "1" (pick Yes) if the displayed box correctly localizes any object of the given class or press "0" (pick No) otherwise.

The definition of a correct bounding box is crucial. We carefully phrase our instruc-

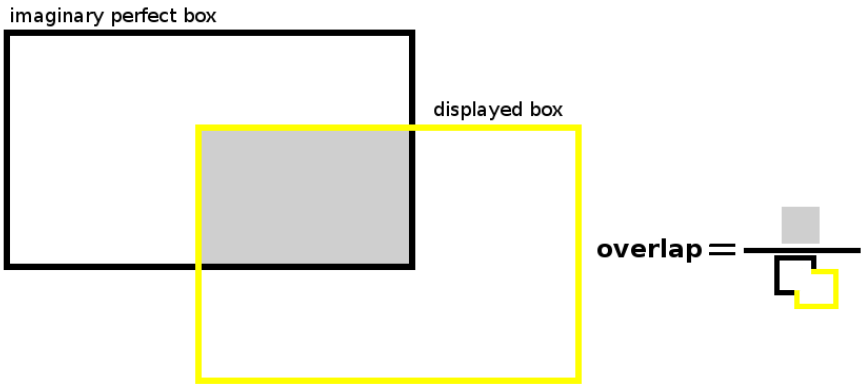


Figure 4.6: *The definition of a correct box according to the IoU criterion [Everingham et al., 2010] as we explain it to the annotators during the instructions of the task.*

tions as: "Imagine a perfectly tight box around the object. The displayed box should be considered as correct if its overlap with the imaginary perfect box is greater than 0.5" (Figure 4.6). We also include examples of perfect ($IoU = 1$), clearly wrong ($IoU < 0.4$) and clearly correct ($IoU > 0.6$) boxes on four different images (Figure 4.7).

In order to let annotators know approximately how long the task will take, we suggest a time of 2s per verification. This is an indicative annotation time that we estimated from a small pilot study.

Qualification test. After reading the instructions, the annotators need to pass a qualification test. A qualification test is a good mechanism for enhancing the quality of crowd-sourcing data and for filtering out bad annotators and spammers [Andriluka et al., 2014; Endres et al., 2010; Johnson and Everingham, 2011; Krause et al., 2013; Russakovsky et al., 2015a; Su et al., 2012]. Some annotators do not pay attention to the instructions or do not even read them. Qualification tests have been successfully used to collect image labels, object bounding boxes, and segmentations for some of the most popular datasets (e.g., COCO [Lin et al., 2014] and Imagenet [Russakovsky et al., 2015a; Su et al., 2012]).

The qualification test is designed to mimic our main task of Yes/No verification. We show the annotator a sequence of 10 different images (of the same class) accompanied with a bounding box and ask them to carry out the Yes/No verification task.

As our qualification examples, we use a small set of images with ground-truth bounding boxes. For image, we first generate a very large number of random windows, and then we sample some of them carefully (around 20) so that their overlap follows a uniform distribution from 0 to 1. This selection is important because we want to

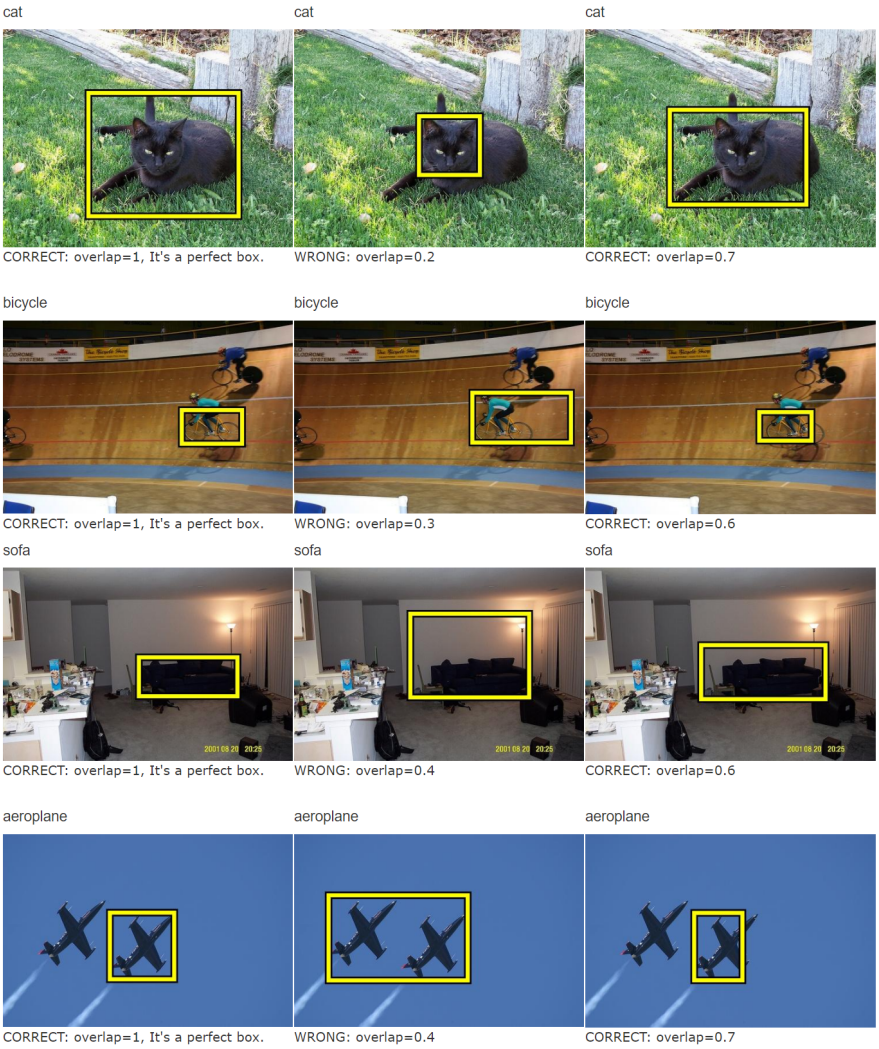


Figure 4.7: ***Instruction examples*** *of perfect ($IoU = 1$), clearly wrong ($IoU < 0.4$) and clearly correct ($IoU > 0.6$) boxes on four images.*

teach the annotators how they should judge boxes from the whole spectrum of overlap values.

Feedback. After the annotators finish the qualification test, they receive a detailed feedback page with all bounding boxes they annotated (Figure 4.8). For each image, we display the bounding box that they annotated, their decision (Yes/No), the real overlap of the box and a simple message about the correctness of their decision.

Success or failure. The annotators pass the qualification test if all their response on examples with $IoU < 0.4$ or $IoU > 0.6$ are correct. We do not penalize annotators if they respond incorrectly on any boundary cases (IoU ≈ 0.5, practically we use $0.4 < IoU < 0.6$). Those that pass the test are recorded as qualified annotators and can

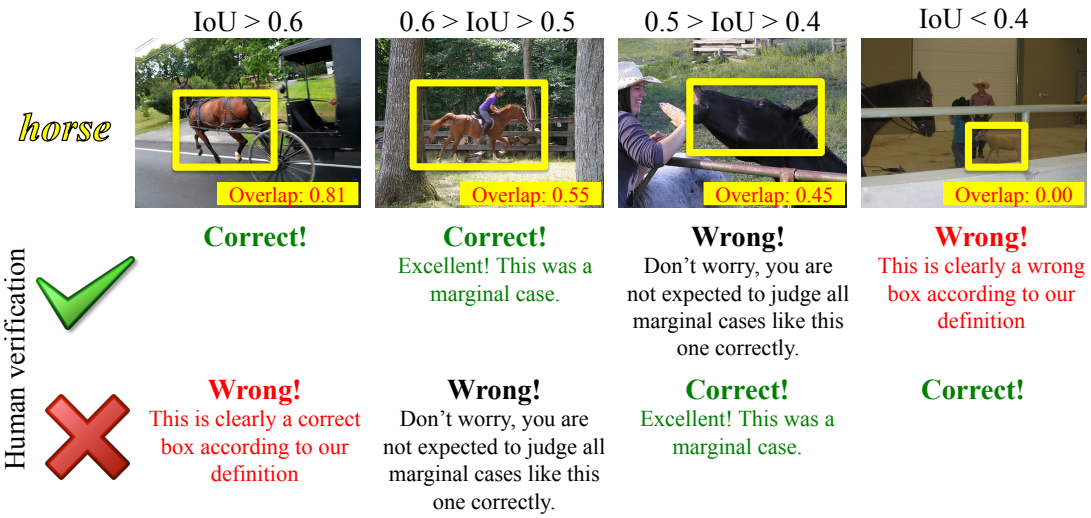


Figure 4.8: *Examples of annotator feedback. For each example, we provide the bounding box they verified, their decision (Yes/No), the real overlap of the box and a simple message about the correctness of their decision. We provide here examples of all possible cases of different annotators decisions and different box overlaps.*

proceed to the main annotation stage. A qualified annotator never has to retake the qualification test. In case of failure, annotators are allowed to repeat the test as many times as they want. The combination of automatically providing rich feedback and allowing annotators to repeat the test makes the training stage interactive and highly effective. Annotators that have reached the desired level of quality can be expected to keep it throughout the annotation [Hata et al., 2017].

Verifying boxes. In the main verification stage, annotators are presented small batches of 20 consecutive images with displayed boxes to verify. For increased efficiency, our batches consist of a single object class. Thanks to this, annotators do not have to re-read the class name for every image, and can keep their mind focused on their prior knowledge about the class to find it rapidly in the image [Torralba et al., 2006]. More generally, it avoids task-switching which is well-known to increase response time and decrease accuracy [Rubinstein et al., 2001; Monsell, 2003].

Quality control. Quality control is a common process when crowd-sourcing image annotations [Bearman et al., 2016; Kovashka and Grauman, 2015; Lin et al., 2014; Russakovsky et al., 2015a; Russell et al., 2008; Sorokin and Forsyth, 2008; Su et al., 2012; Vondrick et al., 2013; Welinder et al., 2010]. We control the quality of the annotation by hiding two evaluation images inside a 20-image batch, and monitor the anno-

tator's accuracy on them. For each evaluation image, we have ground-truth bounding boxes and we generate a large set of random windows similarly to the qualification test. This leads to a large set of golden questions with different boxes at various overlap values for the same evaluation image. We point out that we use extremely few evaluation images. On PASCAL VOC 2007, we used only 40, which amounts to 0.5% of the dataset. This is a negligible overhead. Annotators that fail to respond correctly on these evaluation images are not able to submit the task. Note that we do not use as golden questions any boundary verification cases ($0.4 < IoU < 0.6$). We do not do any post-processing rejection of the submitted data.

4.6 Experimental Results

4.6.1 Dataset and evaluation protocol

PASCAL VOC 2007. We perform experiments on PASCAL VOC 2007 [Everingham et al., 2010], which consists of 20 classes. The trainval set contains 5011 images, while the test set contains 4952 images. We use the trainval set with accompanying image-level labels to train object detectors, and measure their performance on the test set. Following the common protocol for WSOL experiments [Cinbis et al., 2014, 2016; Deselaers et al., 2010; Russakovsky et al., 2012; Wang et al., 2015], we exclude trainval images that contain only difficult and truncated instances, ending up with 3550 images. In Sections 4.6.2, 4.6.3 we carry out a detailed analysis of our system in these settings, using AlexNet as CNN architecture [Krizhevsky et al., 2012] and using expert annotators. In Section 4.6.4 we also present results when using the complete trainval set, both expert and naive human verification crowd-sourced on Amazon Mechanical Turk (AMT), and the deeper VGG16 as CNN architecture [Simonyan and Zisserman, 2015].

Evaluation. Given a training set with image-level labels, our goal is to localize the object instances in this set and to train good object detectors, while minimizing human annotation effort. We evaluate this by exploring the trade-off between localization performance and quality of the object detectors versus required annotation effort. We quantify localization performance in the training set with the Correct Localization (CorLoc) measure [Bilen et al., 2014, 2015; Cinbis et al., 2014, 2016; Deselaers et al., 2010; Russakovsky et al., 2012; Siva and Xiang, 2011; Wang et al., 2015; Bilen

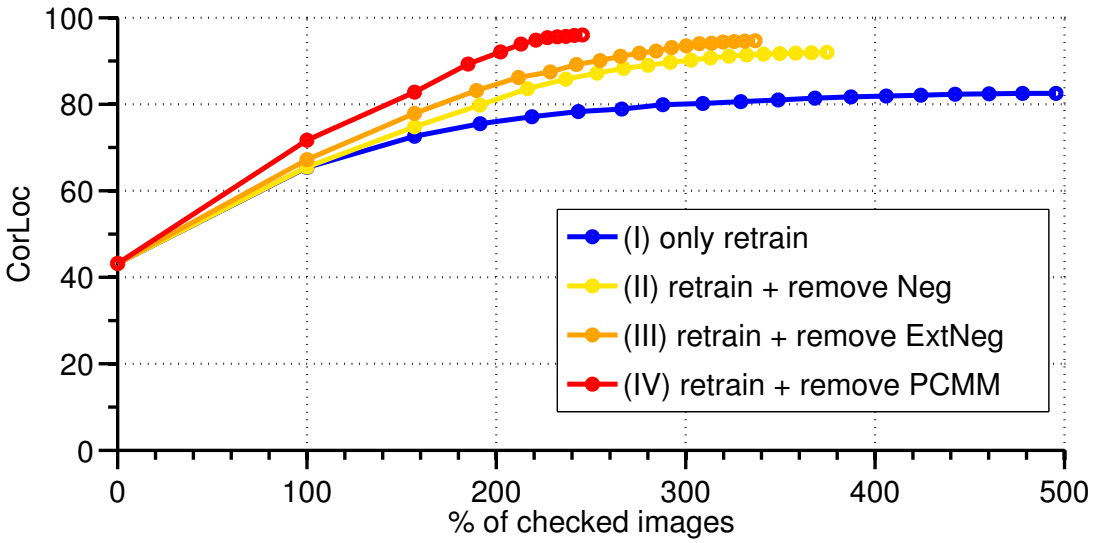


Figure 4.9: *Trade-off between the number of verifications and CorLoc for the simulated verification case on PASCAL VOC 2007.*

and Vedaldi, 2016]. CorLoc is the percentage of images in which the bounding-box returned by the algorithm correctly localizes an object of the target class (i.e., IoU ≥ 0.5).

We quantify object detection performance on the test set using mean average precision (mAP), as standard in PASCAL VOC 2007. We quantify annotation effort both in terms of the number of verifications and in terms of actual human time measurements.

As most previous WSOL methods [Bilen et al., 2014, 2015; Cinbis et al., 2014, 2016; Deselaers et al., 2010; Russakovsky et al., 2012; Siva and Xiang, 2011; Song et al., 2014a,b; Wang et al., 2015], our scheme returns exactly one bounding-box per class per training image. This enables clean comparisons to previous work in terms of CorLoc on the training set, and keeps the human verification tasks simple (as we do not need to ask the annotators whether they see additional instances in an image). Note how at test time the detector is capable of localizing multiple objects of the same class in the same image (and this is captured in the mAP measure).

Compared methods. We compare our approach to the fully supervised alternative by training the same object detector (Section 4.4.1) on the same training images, but with manual bounding-boxes (again, one bounding-box per class per image). On the other end of the supervision spectrum, we also compare to a modern MIL-based WSOL technique run on the same training images, but without human verification (Section 4.4.2). Since that technique also forms the initialization step of our method, this comparison

reveals how much farther we can go with human verification.

For MIL WSOL, the effort to draw bounding-boxes is zero. For fully supervised learning we take the actual annotation times for ILSVRC [Russakovsky et al., 2015a] from Su et al. [2012]: they report 35 seconds for drawing and verifying one high-quality bounding-box. These timings are also representative for PASCAL VOC, since it is of comparable difficulty and its annotations are of comparable quality. The bounding-boxes in both datasets are of high quality and precisely match the object extent.

4.6.2 Simulated verification

We first use simulated verification to determine how best to use the verification signal. We simulate human verification by using the available ground-truth bounding boxes. Note how these are *not* given to the learning algorithm, they are only used to derive the verification signals of Section 4.3.1. Figure 4.9 compares four ways to use the verification signal in terms of the trade-off between the number of verifications and CorLoc (Section 4.3.3): **(I) only retrain** the object detector (using positively verified detections D_n^+); **(II) retrain + remove Neg**: for Yes/No verification, retrain and reduce the search space by eliminating one proposal for each negatively verified detection; **(III) retrain + remove ExtNeg**: for Yes/No verification, retrain and eliminate all proposals overlapping with a negatively verified detection; **(IV) retrain + remove PCMM**: for YPCMM verification, retrain and eliminate proposals according to the type of error.

As Figure 4.9 shows, even using verification just to re-train the object detector (I) drastically boosts CorLoc from the initial 43% (achieved by MIL WSOL) up to 82%. This requires checking each training image on average 4 times. Using the verification signal in the re-localization step by reducing the search space (II–IV) helps to reach this CorLoc substantially faster (1.6–2 checks per image). Moreover, the final CorLoc is much higher when we reduce the search space. Removing negatively verified detections brings a considerable jump to 92% CorLoc (II); removing all proposals around negatively verified detections further increases CorLoc to 95% (III); the Yes/Part/Container/Mixed/Missed strategy appears to be the most promising, achieving a near-perfect 96% CorLoc after checking each image only 2.5 times on average. These results show that feeding the verification signal into both the re-training and re-localization steps quickly results in a large number of correctly located object instances.

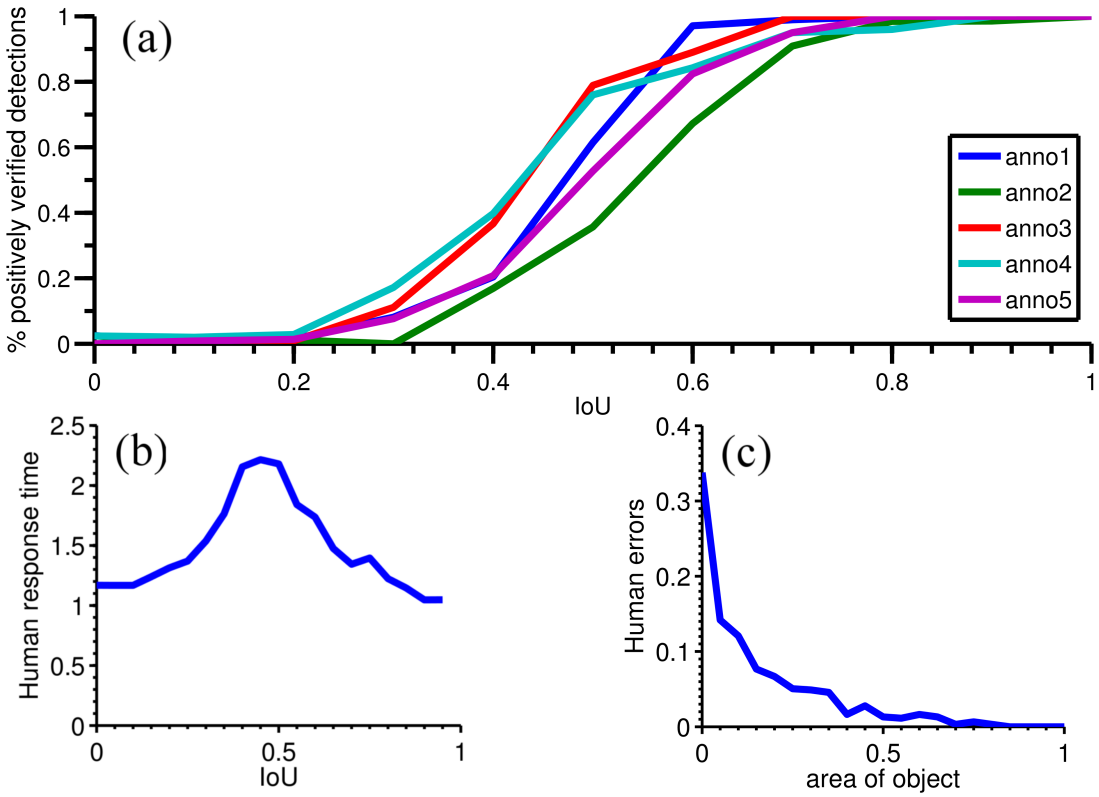


Figure 4.10: ***Analysis of expert verification.*** *(a) Percentage of positively verified detections as a function of ground-truth IoU, for each annotator. (b) Average human response time as a function of ground-truth IoU. (c) Percentage of incorrectly verified detections as a function of object area (relative to image area).*

Figure 4.4 compares the search process of Yes/No and YPCMM verification strategies on a bird example. The second detection is diagnosed in YPCMM as a container. This focuses the search to that particular part of the image. In contrast, detections of the Yes/No case jump around before finding the detection. This shows that the search process of YPCMM is more targeted. However, in both cases the target object location is found rather quickly.

In conclusion, YPCMM is the most promising verification strategy, followed by Yes/No with removing all proposals overlapping with a negatively verified detection (III). Since Yes/No verification is intuitively easier and faster, we try both strategies in experiments with expert human annotators.

4.6.3 Expert human verification

In this section, we use human verification from our expert annotators (Section 4.5.1).

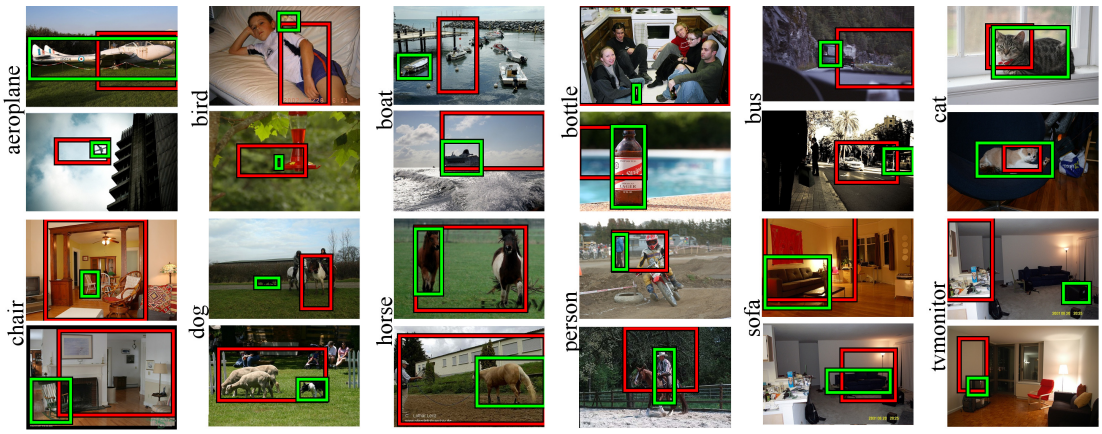


Figure 4.11: *Examples of objects localized by using our proposed Yes/No expert verification scheme on the trainval set of PASCAL VOC 2007 (Section 4.6.3). For each example, we compare the final output of our scheme (green box) to the output of the reference multiple instance learning (MIL) weakly-supervised object localization approach (red box) (Section 4.4.2).*

Annotation time. The mean response time of the Yes/No verification task was 1.6 seconds per verification. The more elaborate task of YPCMM verification took on average 2.4 seconds per verification.

Analysis of expert verification. We evaluate here the performance of expert annotators to the Yes/No verification task. Figure 4.10(a) reports the percentage of positively verified detections as a function of their IoU with the ground-truth bounding-box. We observe that experts behave quite closely to the desired PASCAL criterion (i.e. IoU > 0.5) which we use in our simulations. All expert annotators behave identically on easy cases (IoU < 0.25, IoU > 0.75). On boundary cases (IoU ≈ 0.5) we observe some annotator bias. For example, *anno*2 tends to judge boundary cases as wrong detections, whereas *anno*3 and *anno*4 judge them more frequently as correct. Overall the percentage of incorrect Yes and No judgments are 14.8% and 8.5%, respectively. Therefore there is a slight bias towards Yes, i.e. humans tend to be slightly more lenient than the IoU> 0.5 criterion.

While the average human response time is 1.6 s for the Yes/No verification, the response time for verifying difficult detections (IoU ≈ 0.5) is significantly higher (2.2 s, Figure 4.10(b)). This shows how the difficulty of the verification task is directly linked to the IoU of the detection, and is reflected in the time measurements. We also found that expert verification errors strongly correlate with the area of objects: 48% of all errors are made when objects occupy less than 10% of the image area (Figure 4.10c).

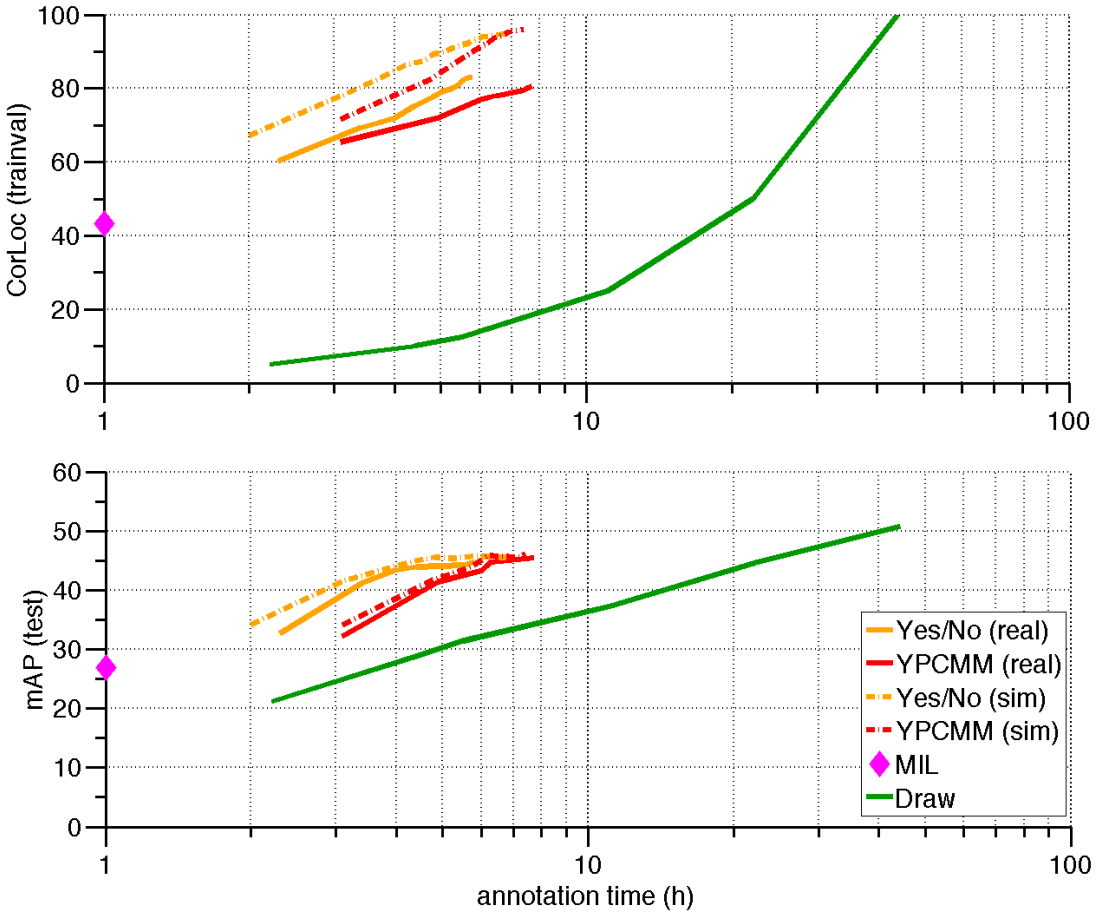


Figure 4.12: *Evaluation on PASCAL VOC 2007: CorLoc and mAP against human annotation time in hours (log-scale). All orange and red curves are variations of our proposed scheme, with simulated ('sim') and expert real ('real') annotators. 'Draw' indicates learning from manual bounding-boxes (full supervision). 'MIL' indicates learning under weak supervision only, without human verification (sec. 4.4.2). The fully supervised approach needs* 8× *extra annotation time to obtain similar performance to our framework.*

Simulated vs. expert annotators. We first compare simulated and expert human annotators by plotting CorLoc and mAP against actual annotation time (rather than as number of verifications as in Section 4.6.2). For simulated verification, we use average expert annotation time as reported above. Figure 4.12 shows the results on a log-scale. While expert annotators are somewhat worse than simulations in terms of CorLoc on the training set, the mAP of the resulting object detectors on the test set are comparable. The diverging results for CorLoc and mAP is because expert human judgment errors are generally made on boundary cases with bounding-boxes that only approximately cover an object (Figure 4.10(a)). Using these cases either as positive or

negative training examples, the object detector remains equally strong. To conclude, in terms of training high quality object detectors, actual expert annotators reliably deliver similar results as simulated annotators.

In Section 4.6.2 we observed that YPCMM needs fewer verifications than Yes/No. However, in terms of total annotation time, the Yes/No task has the more favorable trade-off: Yes/No achieves 83% CorLoc and 45% mAP by taking 5.8 hours of annotation time, while YPCMM achieves 81% CorLoc and 45% mAP by taking 7.7 hours (Figure 4.12). Hence we conclude that the Yes/No task is preferable for human annotation, as it is easier and faster.

Weak supervision vs. expert verification. We now compare the reference MIL WSOL approach that we use to initialize our process (Section 4.4.2 and magenta diamond in Figure 4.12) to the final output of our Yes/No expert human verification scheme (solid orange line, Figure 4.12). While MIL WSOL achieves 43% CorLoc and 27% mAP, using expert verification bring a massive jump in performance to 83% CorLoc and 45% mAP. Hence at a modest cost of 5.8 hours of annotation time we achieve substantial performance gains. Examples in Figure 4.11 show that our approach localizes objects more accurately and succeeds in more challenging conditions, e.g. when the object is very small and appears in a cluttered scene.

Full supervision vs. expert verification. We now compare our Yes/No expert verification scheme (solid orange line, Figure 4.12) to standard fully supervised learning with manual bounding-boxes (solid green lines). The object detectors learned by our scheme achieve 45% mAP, almost as good as the fully supervised ones (51% mAP). Importantly, fully supervised training needs 44 hours of annotation time, when assuming an optimistic 35 s per box). Our method instead requires only 5.8 hours, a reduction in human effort of 8×.

From a different perspective, when given the same annotation time as our approach (5.8 hours), the fully supervised detector only achieves 32% mAP.

We conclude that by the use of just an inexpensive verification component, we can train strong object detectors at little cost. This is significant since it enables the cheap creation of high quality object detectors for a large variety of classes, reducing the need for massive annotation efforts such as ImageNet [Russakovsky et al., 2015a].

4.6.4 Complete training set, AMT human verification and deeper network

In the previous section, we showed that in terms of training high quality object detectors, expert annotators reliably deliver similar results as simulated annotators. In this section, we perform experiments with both expert and AMT annotators and we discuss the differences in their performance on our verification task

Dataset. In this section, we perform here experiments using the complete trainval set of PASCAL VOC 2007 (i.e. 5011 images). We test the detectors again on the test set and we follow the exact evaluation settings explained in the Section 4.6.1.

Annotation time of AMT annotators. The mean response time of AMT annotators was 1.8 seconds per verification. Interestingly, this time is only slightly above the annotation time of expert annotators (1.6 s per verification).

Analysis of AMT annotators. We perform here an analysis of the Yes/No human verification results which we collected using our crowd-sourced framework of Section 4.5.2. In Figure 4.13, we report the percentage of positively verified detections as a function of their IoU with the ground-truth bounding-box (blue line). As expected, the behavior of AMT annotators is slightly worse that the expert behavior. The AMT annotators perform quite well and on par with experts on the extreme cases $IoU > 0.6$ and $IoU < 0.2$. However, we observe a tendency on judging boundary and slightly bad ($0.3 < IoU < 0.5$) boxes as correct.

Cost. We paid annotators \$0.08 to annotate a batch of 20 images. Based on their mean response time this results in a wage of about \$8 per hour. The total cost for annotating the whole trainval set of PASCAL VOC 2007 was \$83.8.

Expert vs. AMT verification. We first compare expert and AMT human annotators by plotting CorLoc and mAP against annotation time. Note that here we re-did our experiment using the complete training set. Results are shown in Figure 4.14. Our expert verification scheme yields 81% CorLoc and 50% mAP using 9.2 hours of annotation. Note that the mAP of expert verification is 5% better than the one reported in the previous section as here we train from the complete trainval set. Our verification scheme using AMT annotators perform slightly worse: 76% CorLoc and 47% mAP. Even though ATM annotators are a bit slower than experts, the total human annota-

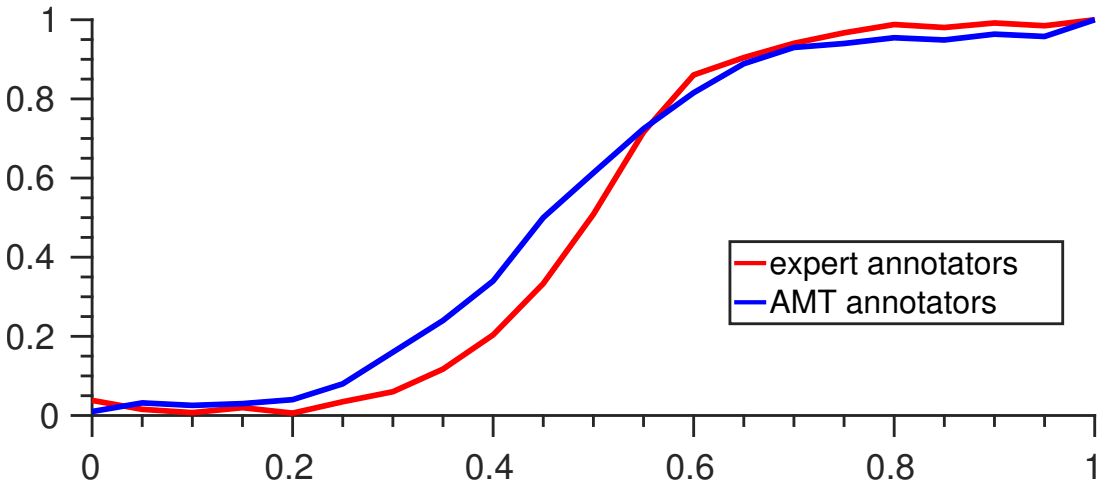


Figure 4.13: ***Comparison between expert and AMT annotators.*** *Percentage of positively verified detections as a function of ground-truth IoU, for AMT annotators (blue line) and expert annotators (red line).*

tion cost using either experts or AMT annotators is about the same. This is because AMT annotators judge more frequently slightly bad boxes as correct (see Figure 4.13) leading to fewer verifications in total that expert verification. Everything below in this section lists CorLoc and mAP for both AMT and expert verification.

Weak supervision vs. AMT verification. MIL WSOL achieves 43% CorLoc and 30% mAP. Note that this mAP result is slightly better than the mAP reported in the previous section as we now train from the complete trainval set. Using human verification bring a massive jump in performance to 76%-81% CorLoc and 47%-50% mAP. Hence at a modest annotation cost we achieve substantial performance gains.

The state-of-the-art WSOL approaches perform as follows when using AlexNet: Cinbis et al. [2016] achieve 52.0% CorLoc and 30.2% mAP, Bilen et al. [2015] achieve 43.7% CorLoc and 27.7% mAP, Wang et al. [2015] achieve 48.5% CorLoc and 31.6% mAP, Bilen and Vedaldi [2016] achieve 54.2% CorLoc and 34.5% mAP.

Our method using human verification substantially outperforms all of them, reaching 76%-81% CorLoc and 47%-50% mAP. Hence at a modest extra annotation cost, we obtain many more correct object locations and train better detectors.

Full supervision vs. AMT human verification. Under full supervision, Girshick [2015] reports 57% mAP based on AlexNet. Training Fast R-CNN from one bounding box per class per image, results in 55% mAP. Our Yes/No human verification scheme

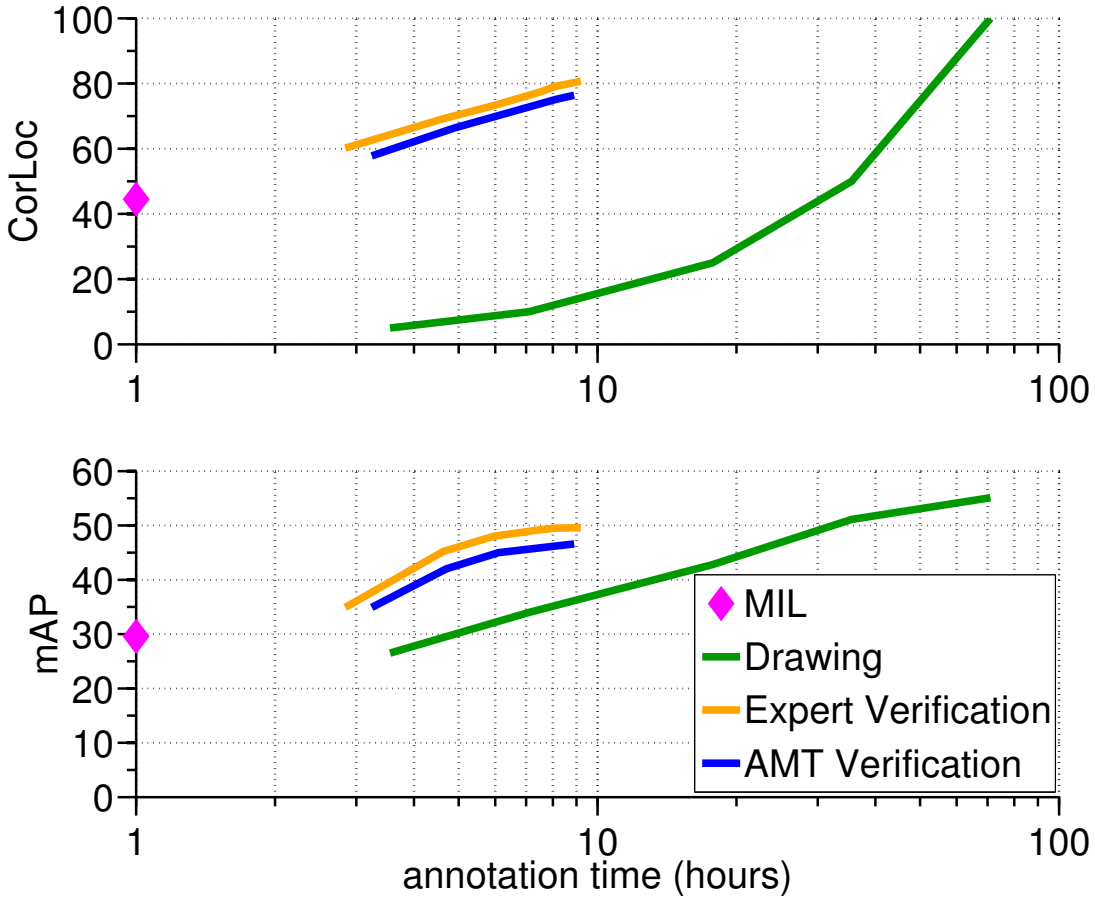


Figure 4.14: *Evaluation on PASCAL VOC 2007 using the complete trainval set: CorLoc and mAP against human annotation time in hours (log-scale).*

gets to 47%-50% mAP.

Deeper CNN Additionally, we experiment with VGG16 instead of AlexNet, with the same settings. Training with full supervision leads to 66% mAP, while our verification scheme delivers 54% mAP with AMT annotators and 58% mAP with experts. Our reference MIL WSOL yields 32% mAP.

Hence, on both CNN architectures our verification-based training scheme using either expert or AMT annotators produces high quality detectors, achieving 80-90% of the mAP of their fully supervised counterparts.

Influence of qualification test and quality control. The above findings indicate that using our designed crowd-sourcing framework we can train naive annotators and make them behave reasonably well on our Yes/No verification task. To better understand the influence of our qualification test and the quality control mechanism, we conducted

	reduced training set		complete training set			
	Yes/No (expert)	Full supervision	Weak supervision	Yes/No (expert)	Yes/No (AMT)	Full Supervision
AlexNet	45%	51%	30%	50%	47%	55%
VGG16	55%	61%	32%	58%	54%	66%

Table 4.1: *Comparison of mAP results between our Yes/No (expert or AMT) human verification scheme, weak supervision and full supervision using different training sets and different network architectures. 'reduced training set': excluding trainval images containing only difficult and truncated instances (3550 images); 'complete training set': all trainval images (5011).*

a series of small-scale crowd-sourcing experiments on 400 images of PASCAL VOC 2007 trainval set. In Figure 4.15, we report once more the percentage of positively verified detections as a function of their IoU with the ground-truth bounding box for all these small-scale experiments. Using a qualification test vastly improves the quality of verifications. We observe that without a qualification test and a quality control mechanism, we obtain quite poor verifications. AMT annotators judge quite frequently clearly wrong boxes ($IoU < 0.3$) as correct and vice versa. Using a qualification test significantly improves the accuracy of the verifications. The quality control also brings a further improvement. Using our full crowd-sourced protocol, we observe that AMT annotators perform quite well and on par with experts on the extreme cases IoU ¿ 0.6 and IoU ¡ 0.2.

4.7 Conclusions

We proposed a scheme for training object class detectors which introduces a human verification step to improve the re-training and re-localization steps common to most weakly supervised approaches. Experiments on PASCAL VOC 2007 with both expert and crowd-sourced annotators show that our scheme produces detectors performing almost as good as those trained in a fully supervised setting, *without ever drawing any bounding-box*. As the verification task is very quick, our scheme reduces the total human annotation time by $8\times$.

One important aspect of our scheme is that one can control the quality of the generated bounding boxes in the training set. Our scheme results in a training set with generated bounding boxes above a desired IoU threshold. In this Chapter, we only considered 0.5 IoU. However, one can simply use stricter threshold values (e.g., 0.7

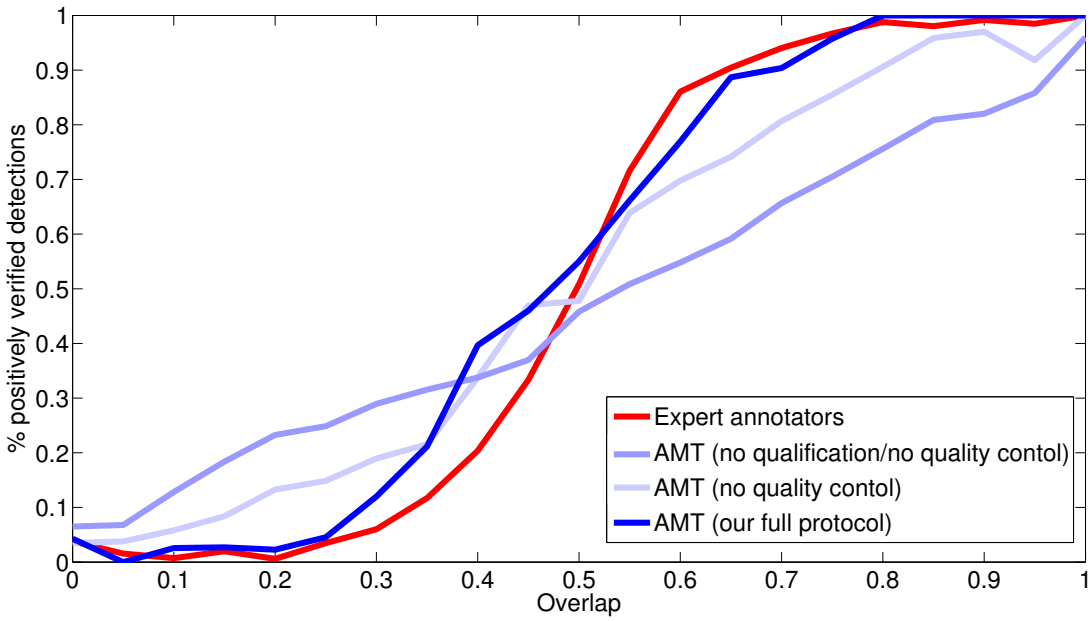


Figure 4.15: *Influence of the qualification test and quality control on the accuracy of AMT verifications (on 400 images from PASCAL VOC 2007).*

IoU) aiming at more accurate bounding boxes that can potentially be used to train higher quality object detectors at a cost of extra human verifications per image.

One limitation of our scheme is that the verifications need to be acquired iteratively during detector training. If the initialization of our scheme (Section 4.4.2) or the underlying object detector (Section 4.4.1) change, we should re-collect all human verifications.

Chapter 5

Training object class detectors with click supervision

Contents

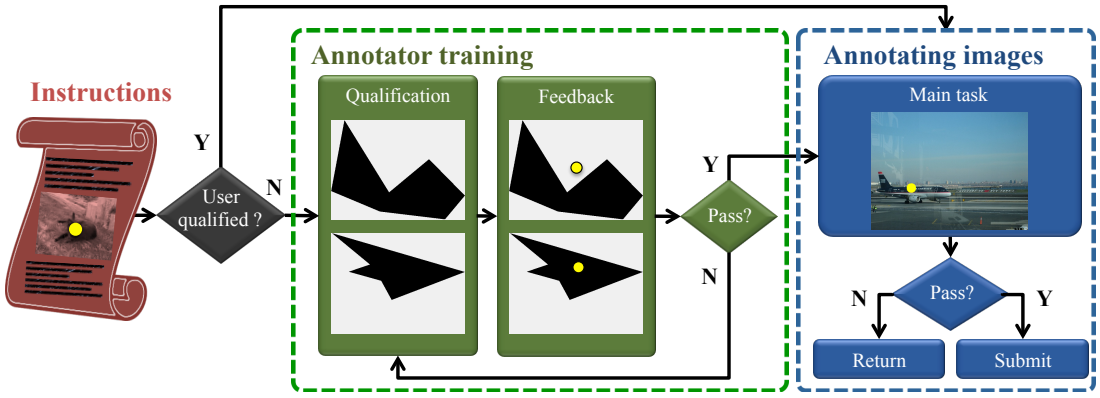


Figure 5.1: ***The workflow of our crowd-sourcing framework for collecting click annotations.*** *The annotators read a set of instructions and then go through an interactive training stage that consists of a simple qualification test based on synthetic polygons. After completing it, they receive a detailed feedback on how well they performed. Annotators who successfully pass the qualification test can proceed to the annotation stage. In case of failure, they can repeat the test as many times as they want.*

5.1 Introduction

In the previous chapter we proposed a scheme for training object detectors using only human verification. In this chapter, we propose another efficient annotation scheme to train object detectors. Once more, we aim to minimize human annotation effort while producing high-quality detectors. To this end we propose annotating objects by asking the annotators to simply click on the object center. Unlike human verifications, clicks can be acquired separately, independently of the detector training framework used. In Section 5.5, we compare this center-clicking scheme with the one presented in the previous chapter.

Clicking on an object can be seen as the human-computer-interaction equivalent of pointing to an object. Pointing is a natural way for humans to communicate that emerges early during cognitive development [Tomasello et al., 2007]. Human pointing behavior is well-understood in human-computer interaction, and can be modeled mathematically [Soukoreff and MacKenzie, 2004]. For the purpose of image annotation, clicking on an object is therefore a natural choice. Clicking offers several advantages

over other ways to annotate bounding boxes: (1) is substantially faster than drawing bounding boxes [Su et al., 2012], (2) requires little instructions or annotator training compared to drawing [Su et al., 2012] or verifying bounding boxes [Papadopoulos et al., 2016; Russakovsky et al., 2015b; Su et al., 2012], because it is a task that comes natural to humans, (3) can be performed using a simple annotation interface (unlike bounding box drawing [Su et al., 2012]), and requires no specialized hardware (unlike eye-tracking data as in our scheme presented in Chapter 3). Note that the scheme we propose does not require a human-in-the-loop setup [Deng et al., 2013; Papadopoulos et al., 2016; Parkash and Parikh, 2012; Vijayanarasimhan and Grauman, 2014; Jain and Grauman, 2016b]: clicks can be acquired separately, independently of the detector training framework used.

Given an image known to contain a certain object class, we ask annotators to click on the center of an imaginary bounding box enclosing the object (*center-click* annotations). These clicks provide reliable anchor points for the full bounding box, as they provide an estimate of its center. Moreover, we can also ask two different annotators to provide center-clicks on the same object. As their errors are independent, we can obtain a more accurate estimate of the object center by averaging their click positions. Interestingly, given the two clicks, we can even estimate the *size* of the object, by exploiting a correlation between the object size and the distance of the click to the true center (error). As the errors are independent, the distance between the two clicks increases with object size enabling to estimate the size of the object based on this distance. As a novel component of our crowd-sourcing protocol, we introduce a stage to train the annotators based on synthetic polygons. This enables generating an arbitrarily large set of training questions without using any manually drawn bounding box. Moreover, we derive models of the annotator error directly from this polygon stage, and use them later to estimate object size in real images.

We incorporate these clicks into a reference Multiple Instance Learning (MIL) framework which was originally designed for weakly supervised object detection (Section 4.4.2). It jointly localizes object bounding boxes over all training images of an object class. It iteratively alternates between re-training the detector and re-localizing objects. We use the center-clicks in the re-localization phase, to promote selecting bounding boxes compatible with the object center and size estimated based on the clicks.

Based on extensive experiments with crowd-sourced center-clicks on Amazon Mechanical Turk for PASCAL VOC 2007 and simulations on MS COCO, we demon-

strate that: (1) our scheme incorporating center-click into MIL delivers better bounding boxes on the training set. In turn, this leads to high-quality detectors, performing substantially better than those produced by weakly supervised techniques, with a modest extra annotation effort (less than 4h on the entire PASCAL VOC 2007 trainval); (2) these detectors in fact perform in a range close to those trained from manually drawn bounding boxes; (3) as the center-click task is very fast, our scheme reduces total annotation time by 9× (two clicks) to 18× (one click); (4) given the same human annotation budget, our scheme outperforms our human verification scheme [Papadopoulos et al., 2016] presented in Chapter 4, which was already very efficient.

5.2 Related work

The work of this Chapter is related to work reducing annotation time for training object detectors. More details about WSOL or other ways to reduce annotation effort for training object class detectors can be found in Sections 2.2, 2.3. We focus here on click supervision approaches that were not covered in Chapter 2.

5.2.1 Click supervision

Click annotation schemes have been used in part-based detection to annotate part locations of an object [Branson et al., 2011; Wah et al., 2011], and in human pose estimation to annotate key-points of human body parts [Johnson and Everingham, 2010; Ramanan, 2006; Sapp and Taskar, 2013].

Click supervision has also been used to reduce the annotation time for semantic segmentation [Bearman et al., 2016; Jain and Grauman, 2016a; Bell et al., 2015; Wang et al., 2014b]. Recently, Bearman et al. [2016] collected clicks by asking the annotators to click anywhere on a target object. In Section 5.5.1, we show that our center-click annotations outperforms these click-anywhere annotations for object class detection. Finally, Mettes et al. [2016] proposed to annotate actions in videos with click annotations.

Our work also offers other new elements over the above approaches, e.g. estimating object area from two clicks and training annotators with synthetic polygons.

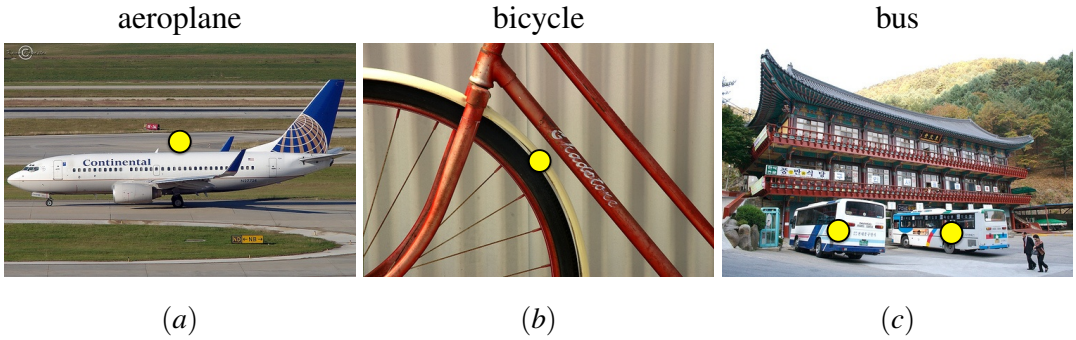


Figure 5.2: ***Instruction Examples****: (a) the desired box center may not be on the object, (b) if the object instance is truncated, click on the center of the visible part and (c) if multiple instances are present, click on the center of any one of them.*

5.3 Crowd-sourcing clicks

We now describe the main components of our crowd-sourcing workflow, which is illustrated in Fig. 5.1.

5.3.1 Instructions

Our annotators are given an image and the name of the target class. Unlike Bearman et al. [2016] where annotators are asked to click anywhere on a target object, we want them to click on the center of an imaginary bounding box around the object (Figure 5.2). This definition of center is crucial, as it provides a strong anchor point for the actual bounding box location. However, humans have a tendency to click on the center of mass of the object, which gives a less precise anchor point for the box location. We therefore carefully phrase our instructions as: "imagine a perfectly tight rectangular box around the object and then click as close as possible to the center of this imaginary box". For concave objects, the box center might even lie outside the object (Figure 5.2(a)).

We also include explanations for special cases: If an object is truncated (i.e. only part of it is visible), the annotator should click on the center of the visible part (Figure 5.2(b)). If there are multiple instances of the target class, one should click on the center of only one of them (Figure 5.2(c)).

In order to let annotators know approximately how long the task will take, we suggest a time of 3s per click. This is an upper bound on the expected annotation time that we estimated from a small pilot study.

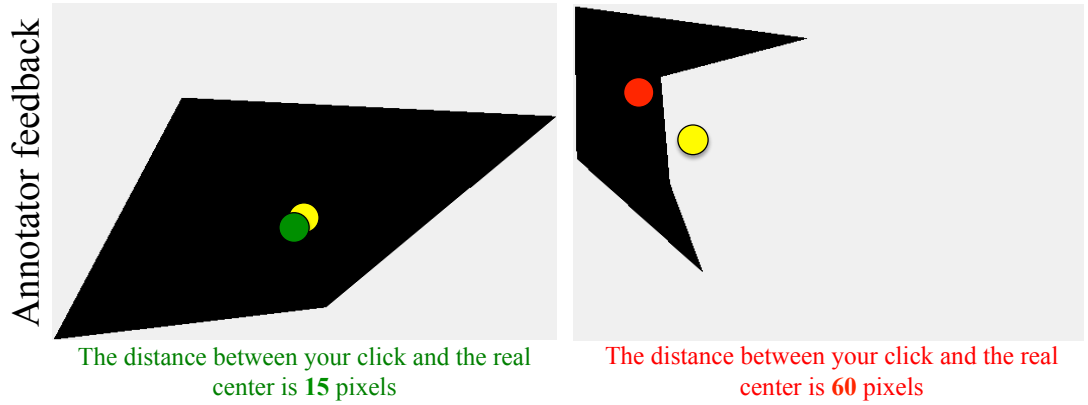


Figure 5.3: *Examples that the annotators receive as feedback. For each example, we provide the real center of the polygon (yellow dot), their click (green or red dot) and the Euclidean distance between the two.*

5.3.2 Annotator training

After reading the instructions, the annotators go through the training stage. They complete a simple qualification test, at the end of which we provide detailed feedback on how well they performed. Annotators who successfully pass this test can proceed to the annotation stage. In case of failure, annotators can repeat the test until they succeed.

Qualification test. Qualification tests have been successfully used for enhancing the quality of the crowd-sourced data and filtering out bad annotators and spammers [Andriluka et al., 2014; Endres et al., 2010; Johnson and Everingham, 2011; Krause et al., 2013; Russakovsky et al., 2015a; Su et al., 2012]. This is necessary because some annotators pay little to no attention to the task instructions.

During a qualification test, the annotator is asked to respond on some questions for which the answers are known. This typically requires experts to annotate a batch of examples (in our case draw object bounding boxes). Instead, we use an annotation-free qualification test in which the annotators need to click on the center of 20 synthetically generated polygons, like the ones in Figure 5.1. Using synthetic polygons allows us to generate an arbitrarily large set of qualification questions with zero human annotation cost. Additionally, annotators cannot overfit to qualification questions or cheat by sharing answers, which is possible when the number of qualification questions is small.

Why polygons? We use polygons instead of axis-aligned rectangles in order to train the annotators on the difference between the center of mass of an object and the center of the imaginary box enclosing the object. Moreover, polygons provide a more real-

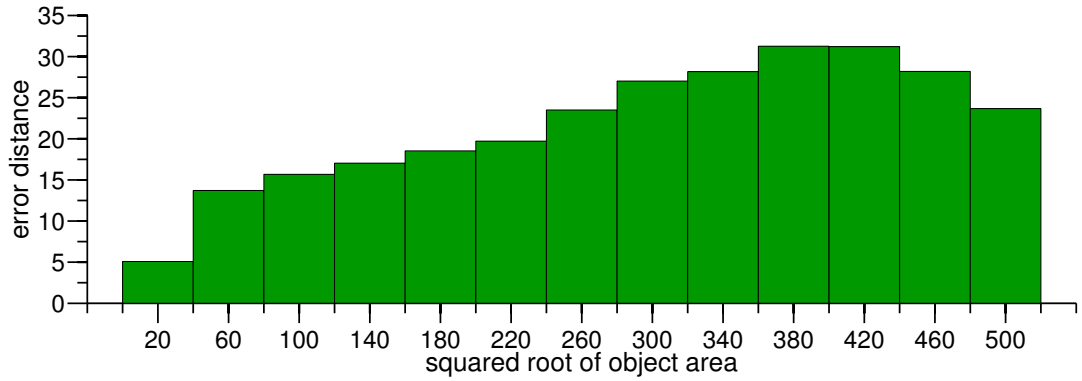


Figure 5.4: *The error distance of the annotators as a function of the square root of the object area.*

istic level of difficulty for the qualification test. Finding the center of an axis-aligned rectangle is trivial, whereas finding the center of a polygon is analogous to finding the center of a real object. And yet, polygons are abstractions of real objects, thus reducing the cognitive load on the annotators, potentially making the training stage more efficient.

Feedback. After the annotators finish the qualification test, they receive a feedback page with all polygon examples they annotated (Fig. 5.3). For each polygon, we display (a) the position of the real center, (b) the position of the annotator's click, and (c) the Euclidean distance in pixels between the two (error distance).

Success or failure. The annotator needs to click close to the real centers of the polygons in order to pass the test. The exact criterion to pass the test is to have an error distance below 20 pixels, on average over all polygons in the test.

The annotators that pass the qualification test are flagged as *qualified annotators* and can proceed to the main annotation task where they work on real images. A qualified annotator never has to retake the qualification test. In case of failure, annotators are allowed to repeat the test as many times as they want until they pass it successfully.

The combination of providing rich feedback and allowing annotators to repeat the test results in an interactive and highly effective training stage.

5.3.3 Annotating images

In the annotation stage, annotators are presented small batches of 20 consecutive images to annotate. For increased efficiency, our batches consist of a single object

Qualification test	Quality control	Error distance
No	No	43.8
images	No	29.4
polygons	No	29.3
polygons	Yes	21.2

Table 5.1: *The influence of the two main elements of our crowd-sourcing protocol on click accuracy.*

class(for more details see Section 4.5.2-Verifying boxes).

Quality control. Quality control is a common process when crowd-sourcing image annotations [Bearman et al., 2016; Kovashka and Grauman, 2015; Lin et al., 2014; Russakovsky et al., 2015a; Russell et al., 2008; Sorokin and Forsyth, 2008; Su et al., 2012; Vondrick et al., 2013; Welinder et al., 2010]. We control the quality of click annotation by hiding two evaluation images for which we have ground-truth bounding boxes inside a 20-image batch, and monitor the annotator's accuracy on them (golden questions). Annotators that fail to achieve an accuracy above the threshold set in the qualification test are not able to submit the task. We do not do any post-processing of the submitted data.

We point out that we use extremely few golden questions, and add them repeatedly to many batches. On PASCAL VOC 2007, we used only 40, which amounts to 0.5% of the dataset. This is a negligible overhead.

5.3.4 Data collection

We implemented our annotation scheme on Amazon Mechanical Turk (AMT) and we collected click annotations for all 20 classes of the whole trainval set of PASCAL VOC 2007 [Everingham et al., 2010]. Each image was annotated with a click by two different annotators for each class present in the image. This results in 14,612 clicks in total for the 5,011 trainval images.

Annotation time. During the annotation stage we measure the annotator's response time from the moment the image appears until they click. The mean response time was 1.87s. This indicates that the task can be performed very efficiently by annotators. Note that we are able to annotate the whole PASCAL VOC 2007 trainval set with one click per object class per image in only 3.8 hours.

Interestingly, the response time we measured is comparable to image-level annotation time (1.5s in Krishna et al. [2016]) indicating that most of the time is spent on the visual search to find the object and not on clicking on it. This time is also in consistency with our findings in Chapter 3 (1 second to find the object). Also, our requirement to click on the center of the object does not slow down the annotators: our response time is comparable to the time reported in Bearman et al. [2016] for click-anywhere annotations.

We examined the response time as a function of the area of the target object and we observed an interesting phenomenon. Response time does not increase when the object becomes smaller, ranging from 1.7s for very small objects to 2.2s for object as big as the whole image. We hypothesize that while small objects are more difficult to find, estimating their center is easier than for large objects.

Error analysis. We evaluate the accuracy of the collected clicks by measuring their distance from the true centers of the ground-truth object bounding boxes. In Figure 5.4 we show this error distance as a function of the square root of the object area. As expected, the error distance in pixels increases as the object area increases. However, it slightly drops as the object occupies the whole image. This is likely because such images have truncated instances, which means the annotator needs to click in the center of the image rather than the center of the object, an easier task. In general, the error distances are quite low: 19.5 pixels on average with a median of 13.1 pixels (the images are 300x500 on average).

Next, we want to understand the influence of using a qualification test, using quality control, and using polygons or real examples during the qualification test. Therefore we conducted a series of smaller-scale crowd-sourcing experiments on 400 images of PASCAL VOC 2007 trainval. As Table 5.1 shows, using a qualification test reduces average error substantially, from 43.8 to 29.4 pixels. Interestingly, using polygons instead of real examples does not influence the error at all, demonstrating that our proposed qualification test is well-suited to train annotators. Quality control, hiding two evaluation images inside the task of annotating images, brings the error further down to 21.2 pixels (on the full dataset we measure 19.5 pixels error). Finally, we note that all four variants in Table 5.1 resulted in similar annotation time. Hence qualification tests or quality control has no significant influence on the speed of the annotators.

Cost. We paid annotators $0.10 to annotate a batch of 20 images. Based on their mean response time this results in a wage of about $9 per hour. The total cost for annotating

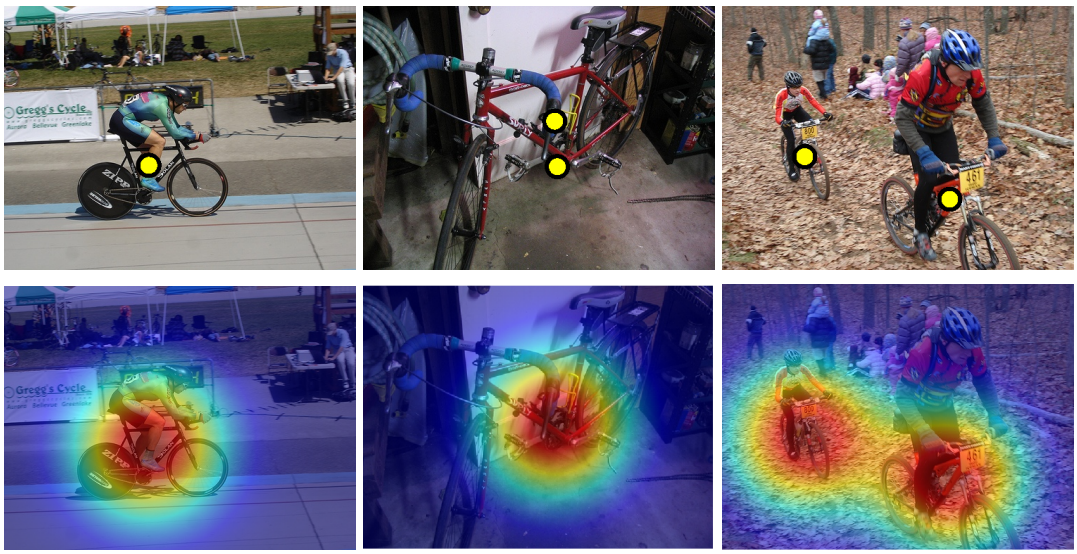


Figure 5.5: ***Box center score*** S_{bc} *on bicycle examples. (left): One-click annotation. (middle): Two-click annotation on the same instance. (right): Two-click annotation on different instances. The values of each pixel in the heatmaps give the* S_{bc} *of an object proposal centered at that pixel.*

the whole trainval set of PASCAL VOC 2007 with two click annotations was $75.40 (or $37.70 for one click annotation).

5.4 Incorporating clicks into WSOL

We now present how we incorporate our click supervision into a reference Multiple Instance Learning (MIL) framework, which is typically used in weakly supervised object detection (WSOL) and was also explained in Section 4.4.2. All explanations in this section consider working with one object class at a time, as we treat them essentially independently.

5.4.1 Reference Multiple Instance Learning (MIL)

The input to MIL is a training set with positive images, which contain the target class, and negative images, which do not. We represent each image as a bag of object proposals extracted using Edge-Boxes [Zitnick and Dollár, 2014]. Following Girshick et al. [2014]; Cinbis et al. [2016]; Bilen et al. [2014, 2015]; Song et al. [2014a]; Wang et al. [2015], we describe each object proposals with a 4096-dimensional feature vector using the Caffe implementation [Jia, 2013] of the AlexNet CNN [Krizhevsky et al.,

2012]. We pre-trained the CNN on the ILSVRC [Russakovsky et al., 2015a] dataset using only image-level labels (no bounding box annotations).

We iteratively build an SVM appearance model $\mathcal{A}$ by alternating between two steps:

(I) *re-localization*: in each positive image, we select the proposal with the highest score given by the current appearance model $\mathcal{A}$.

(II) *re-training*: we re-train the SVM using the current selection of proposals from the positive images, and all proposals from negative images.

As initialization, in the first iteration we train the classifier using complete images as positive training examples [Cinbis et al., 2014, 2016; Pandey and Lazebnik, 2011; Russakovsky et al., 2012; Nguyen et al., 2009; Kim and Torralba, 2009].

Refinements. In order to obtain a competitive baseline, we apply again two refinements to the standard MIL framework. First, we use multi-folding [Cinbis et al., 2016] and second, we combine the score given by the appearance model $\mathcal{A}$ with a general measure of "objectness" [Alexe et al., 2010] O.

Formally, at step (I) we linearly combine the scores $\mathcal{A}$ and O under the assumption of equal weights. The score of each proposal p is given by $S_{ap}(p) = \frac{1}{2} \cdot \mathcal{A}(p) + \frac{1}{2} \cdot O(p)$.

Deep MIL. After MIL converges (typically within 10 iterations), we perform two additional iterations where during the step (II) we deeply re-train the whole CNN network, instead of just an SVM on top of a fixed feature representation. During these iterations we use Fast R-CNN [Girshick, 2015] as the appearance model $\mathcal{A}$.

5.4.2 One-click supervision

Motivation. Click annotations on object centers derived using our crowd-sourcing method of Section 5.3 provide a powerful cue for object position. In this section, we improve the reference MIL framework by using the position of one single click c in each image of the target class.

Box center score S_{bc}. Intuitively, simply selecting the object proposal whose center is closest to the click would fail since annotators are not perfectly accurate. Instead, we introduce a score function S_{bc}, which represents the likelihood of a proposal p covering the object according to its center point c_p and the click c

$$S_{bc}(p; c, \sigma_{bc}) = \mathrm{e}^{-\frac{\|c_p - c\|^2}{2\sigma_{bc}^2}} \tag{5.1}$$

where $\|c_p - c\|$ indicates the Euclidean distance in pixels between c and c_p. The standard deviation σ_{bc} controls how quickly the S_{bc} decreases as c_p gets farther from c (Figure 5.5).

Use in re-localization. We use the box center cue S_{bc} in the re-localization step (I) of MIL (Section 5.4.1). Instead of selecting the proposal with the highest score according to the score function S_{ap} alone, we combine it with S_{bc} with a product: $S_{ap}(p) \cdot S_{bc}(p; c, \sigma_{bc})$. In Sec. 5.5.1 we show that this results in improved re-localization, which in turn leads to better appearance models in the next re-training iteration, and ultimately improves the final MIL outcome.

Use in initialization. We also use the click position to improve the MIL initialization. Instead of initializing the positive training samples from the complete images, we now construct windows centered on the click while at the same time having maximum size without exceeding the image borders. This greatly improves MIL initialization, especially in cases where the position of the click is close to the image borders.

5.4.3 Two-click supervision

Motivation. While using two annotator clicks doubles the total annotation time compared to one click, it allows us to estimate the object center even more accurately. Moreover, we can estimate the object area based on the distance between the two clicks.

Box center score S_{bc}. By averaging the positions of two clicks we can estimate the object center more accurately. We simply replace c in Equation (5.1) with the average of the two clicks c_1 and c_2.

However, in images containing multiple instances of the target class, the two annotators might click on different instances (Figure 5.5, right). To address this, we introduce a distance threshold d_{max} beyond which the clicks are considered to target different instances. In that case, we keep both clicks and use them both in Equation (5.1). Formally, if $\|c_1 - c_2\| > d_{max}$, then for each proposal p we use the nearest of the two click to its center c_p.

Box area score S_{ba}. There is a clear correlation between the area of the object and the click's error distance (Figure 5.4). As errors made by two annotators are indepen-

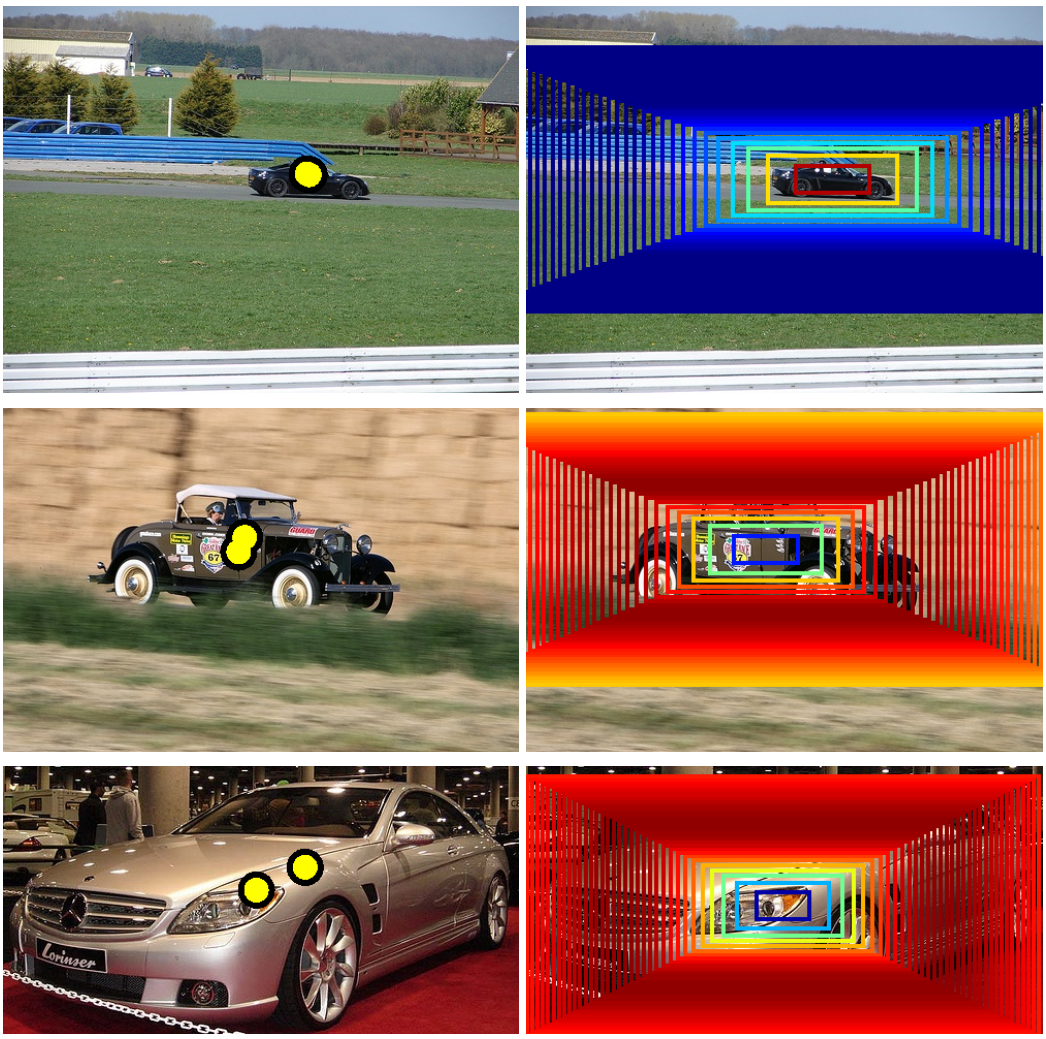


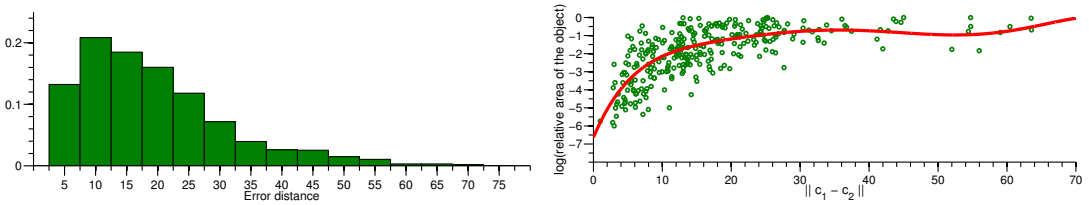
Figure 5.6: ***Box area score*** S_{ba}. *All windows used here have fixed aspect ratio and are centered on the center of the object. The color of the windows shows the value of the* S_{ba} *score. The score is maximal (red) when the area of the proposal matches the estimated object area.*

dent, the distance between their two clicks increases as the object area increases (on average). Therefore we estimate the object area based on the distance between the two clicks c_1 and c_2.

Let $\mu(\|c_1 - c_2\|)$ be a function that estimates the logarithm of the object area (we explain how we learn this function in Sec. 5.4.4). Based on this, for each proposal p we introduce a box area score S_{ba} that represents the likelihood of p covering the object according to the ratio between the proposal area and the estimated object area:

$$S_{ba}(p; c_1, c_2, \sigma_{ba}) = \mathrm{e}^{-\frac{(a_p - \mu(\|c_1 - c_2\|))^2}{2\sigma_{ba}^2}} \tag{5.2}$$

Here a_p is the logarithm of the proposal's area, and $(a_p - \mu)$ indicates the log ratio



(a) (b)

Figure 5.7: *(a) The distribution of errors that the annotators made during our qualification test. (b) The relative area of the synthetic polygons (log scale) as a function of the distance between two clicks. The red line shows the regressed function μ.*

between the two areas. The standard deviation σ_{ba} controls how quickly S_{ba} decreases as a_p grows different from μ.

Figure 5.6 shows an example of the effect of the S_{ba} score on proposals of various areas. For illustration purposes, all proposals used here have a fixed aspect-ratio and are centered on the object. The score is maximal when the area of the proposal matches the estimated object area.

Use in re-localization. We now use all cues in the final score function for a proposal p during the re-localization step (I) of MIL step:

$$S(p) = S_{ap}(p) \cdot S_{bc}(p; c_1, c_2, \sigma_{bc}) \cdot S_{ba}(p; c_1, c_2, \sigma_{ba}) \tag{5.3}$$

5.4.4 Learning score parameters

We exploit the clicks obtained from our qualification task on synthetic polygons to estimate the hyper-parameters of our model: σ_{bc} (Equation (5.1)), d_{max} (Section 5.4.3), σ_{ba} (Equation (5.2)) and the function μ (Equation (5.2)).

Figure 5.7(a) shows the distribution of the annotators' error distances during our qualification test. We estimate σ_{bc} from this distribution. Also, in the same figure we see that the maximum error distance is 70 pixels, hence we set $d_{max} = 70$. Figure 5.7(b) shows the logarithm of the relative area of the synthetic polygons as a function of the distance between the two clicks. We learn the function $\mu(\|c_1 - c_2\|)$ as a polynomial regressor fit to this data (red line in Figure 5.7(b)). Finally, we set σ_{ba} simply as the average error of the area estimation made by the regressor on the polygons.

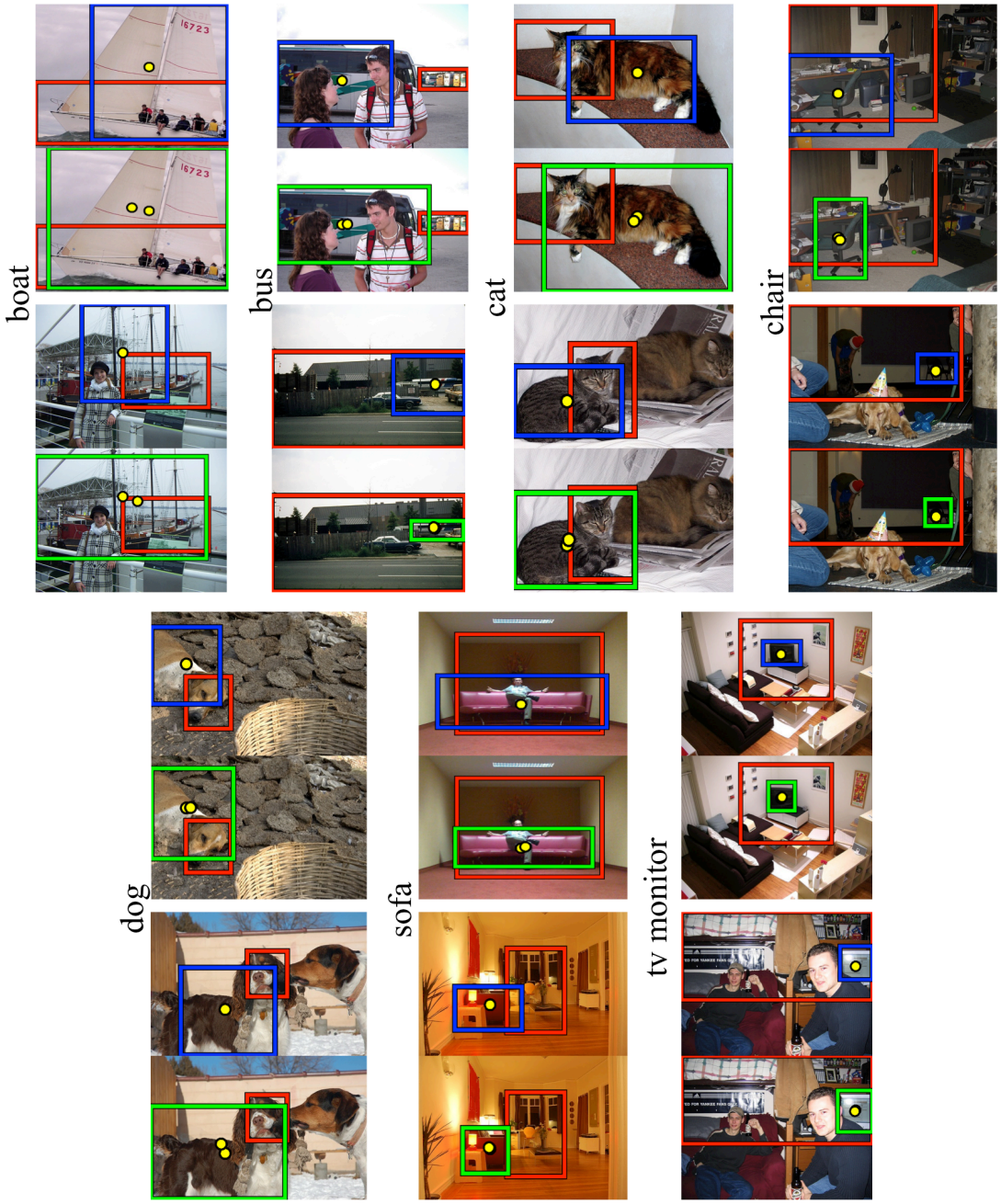


Figure 5.8: *Examples of objects localized on the trainval set of PASCAL VOC 2007 using our one-click (blue) and two-click (green) supervision models. For each example, we also show the localization produced by the reference MIL (red).*

5.5 Experimental results

5.5.1 Results on PASCAL VOC 2007

Dataset. We perform experiments on PASCAL VOC 2007 [Everingham et al., 2010], which has 20 classes, 5,011 training images (trainval), and 4,952 test images. During training we only use image-level labels. Unlike some previous WSOL work which removes images with truncated and difficult objects [Cinbis et al., 2014, 2016; Deselaers et al., 2010; Russakovsky et al., 2012; Wang et al., 2015], we use the complete trainval set.

Object detection model. As object detector we use Fast R-CNN [Girshick, 2015]. Instead of Selective Search [Uijlings et al., 2013] we use EdgeBoxes [Dollar and Zitnick, 2014] as proposals, as they come with an objectness measure [Alexe et al., 2010] which we use inside MIL. Unless stated otherwise, we use AlexNet [Krizhevsky et al., 2012] as the underlying CNN architecture for our method and for all compared methods.

Evaluation. Given a training set with image-level labels (and possibly click annotations), our goal is to localize the object instances in this set and to train good object detectors. We quantify localization performance on the training set with Correct Localization (CorLoc), enabling direct comparison with WSOL methods [Bilen et al., 2014, 2015; Bilen and Vedaldi, 2016; Cinbis et al., 2016; Deselaers et al., 2010; Kantorov et al., 2016; Russakovsky et al., 2012; Siva and Xiang, 2011; Wang et al., 2015]. CorLoc is the percentage of images in which the bounding-box returned by the algorithm correctly localizes an object of the target class (i.e., IoU ≥ 0.5). We measure the performance of the trained object detector on the test set using mean average precision (mAP). We quantify annotation effort in terms of actual human time measurements.

Compared methods. We compare our approach to the fully supervised alternative by training the same object detector [Girshick, 2015] on the same training images, but with manually annotated bounding boxes (one per class per image, for fair comparison). We also compare to a modern MIL-based WSOL technique (Section 5.4.1) run on the same training images, but without click supervision.

For MIL WSOL, the effort to draw bounding boxes is zero. For fully supervised learning we use the actual annotation times for ILSVRC from Su et al. [2012]: 35 seconds for drawing a single bounding box and verifying its quality (Section 2.1.6).

These timings are representative for PASCAL VOC, since their images are of comparable difficulty and quality [Russakovsky et al., 2015a].

We also compare to our human verification scheme [Papadopoulos et al., 2016] presented in Chapter 4, and to various baselines.

Reference MIL. We run the reference MIL WSOL with $k = 10$ folds for 10 iterations, after which it converges. It achieves 43.4% CorLoc on the training set. Applying two deep MIL iterations (Section 5.4.1) on top of this improves to 44.5% CorLoc. The detectors produced by this approach achieve 29.6% mAP on the test set (red dot in Figure 5.9).

One-click supervision yields 73.3% CorLoc. The resulting object detector yields 45.9% mAP (yellow dot in Figure 5.9). Hence, at a modest extra annotation cost of only 3.8 hours we achieve an absolute improvement of +28.8% CorLoc and +16.3% mAP over the reference MIL.

Two-click supervision doubles the annotation time but it improves our model in two ways: (1) we can estimate the object center more accurately, and (2) we can estimate the object area based on the distance between the two clicks. Using the two-click supervision only to improve the box center estimate S_{bc} brings +0.8% CorLoc and +0.9% mAP over using one-click. Including also the box area estimate S_{ba} leads to a total improvement of +5.2% CorLoc and +3.2% mAP over one-click (78.5% CorLoc and 49.1% mAP, orange dot in Figure 5.9). This shows that the box area estimate contributes the most to the improvement brought by two-click over one-click supervision.

State-of-the-art WSOL approaches based on AlexNet architecture [Krizhevsky et al., 2012] perform as follows. Wang et al. [2015]: 48.5% CorLoc and 31.6% mAP. Cinbis et al. [2016]: 52.0% CorLoc and 30.2% mAP. Bilen and Vedaldi [2016]: 54.2% CorLoc and 34.5% mAP. Our two-click supervision outperforms all these methods with 78.5% CorLoc and 49.1% mAP, at a modest extra annotation cost.

Full supervision achieves 55.5% mAP. Our two-click supervision comes remarkably close (49.1% mAP). Importantly, full supervision requires 71 hours of annotation time. Instead, our two-click approach requires only 7.6 hours, a reduction of 9× (or 18× for our one-click approach).

Human verification [Papadopoulos et al., 2016] is shown as the blue line in Fig-

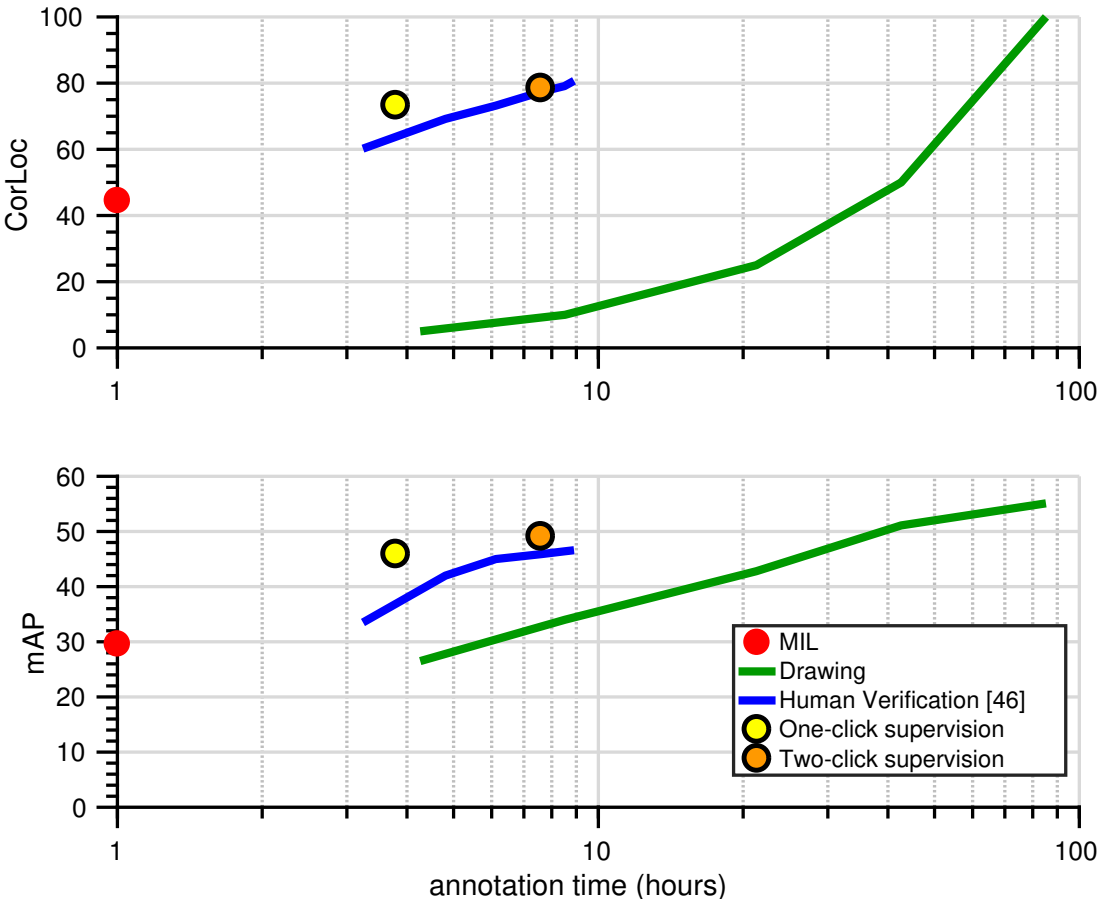


Figure 5.9: ***Evaluation on PASCAL VOC 2007.*** *CorLoc and mAP performance against human annotation time in hours(log-scale).*

ure 5.9. Because the center-click annotations we use in this chapter were crowd-sourced on Amazon Mechanical Turk, for a fair comparison, we use the crowd-sourced human verification results of Section 4.6.4. Given the same annotation time, the one-click supervision is clearly better than human verification scheme. When we use two-click annotations, given the same annotation effort we achieve slightly higher CorLoc and mAP.

Deeper CNN. When using VGG16 [Simonyan and Zisserman, 2015] instead of AlexNet, the fully supervised training leads to 65.9% mAP. Our two-click model achieves 57.5% mAP, while the reference MIL WSOL delivers 32.4% mAP.

Effect of click accuracy. We compare the center-click annotations we collected (Section 5.3) to three alternatives: *(oracle clicks)*: use the centers of the ground-truth boxes as clicks; *(random clicks)*: uniformly sample a pixel inside a ground-truth box; *(click-anywhere)*: we simulate a scenario where humans are instructed to click anywhere on

the object, by mimicking the distribution of the publicly available click annotations of Bearman et al. [2016] on PASCAL VOC 2012. We measure the distances from the centers of the ground-truth boxes to their clicks. Then we build a regressor to predict this distance based on the area of the object. Finally, we apply this regressor on VOC 2007 and displace the ground-truth object centers by the predicted distance.

For simplicity we use the alternative clicks in our one-click supervision model (Section 5.4.2) in one additional re-localization iteration at the end of the reference MIL (as opposed to using it in every iteration). For each of the three alternatives, we use the oracle best value of the parameter σ_{bc}, while for our center-click annotations we use the one learned on the synthetic polygons (Section 5.4.4). As a reference, when used on top of MIL this way, our center-clicks lead to 67.2% CorLoc. Oracle clicks give an upper bound of 73.7% CorLoc, while random clicks on the object do not improve over MIL (43.4% CorLoc). Finally, the click-anywhere scenario achieves 55.5% CorLoc. Interestingly, using our center-clicks leads to +11.7% CorLoc, which shows that they convey more information.

5.5.2 Results on MS COCO

Dataset. The MS COCO dataset [Lin et al., 2014] is more difficult than PASCAL VOC, as demonstrated in Lin et al. [2014], featuring smaller objects on average, and also more object classes (80). We use exactly the same evaluation setup as for PASCAL VOC 2007 and evaluate CorLoc on the training set (82,783 images) and mAP on the val set (40,504 images).

To confirm that MS COCO is significantly more difficult than PASCAL VOC [Lin et al., 2014], in Figure 5.10 we show the distribution of the square root of the relative object area for the two datasets. MS COCO contains substantially more very small objects. For example, 32% of MS COCO objects occupy less than 1% of the image area, while in PASCAL VOC 2007 this is true for only 5% of the objects.

Reference MIL. The reference MIL WSOL achieves 24.2% CorLoc and 8.9% mAP (red dot in Figure 5.11). This is considerably lower than its performance on PASCAL VOC 2007. Deselaers et al. [2010] also observed that the WSOL performance consistently decreases from easy datasets with mostly big objects to hard datasets with cluttered images with small objects.

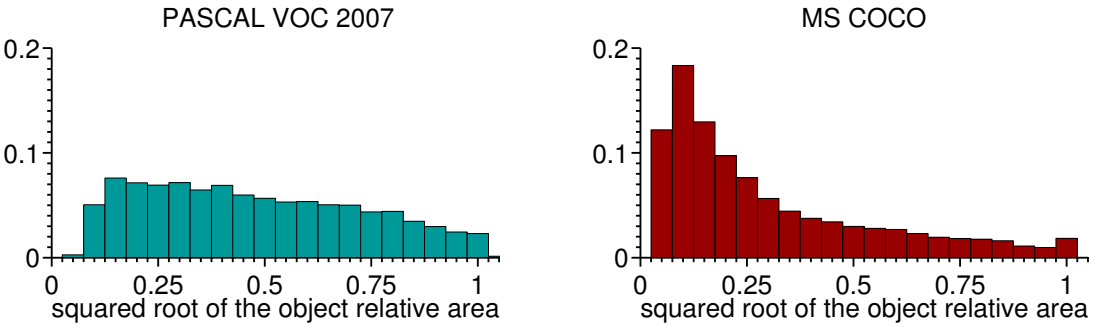


Figure 5.10: *The distribution of the square root of the relative object area on the trainval set of PASCAL VOC 2007 (left) and on the training set of MS COCO (right).*

Click supervision. We did not collect real click annotations for COCO, but instead simulated them. As we want to create a realistic scenario close to real annotators clicks, we did not use the centers of the available ground-truth boxes as simulated clicks. Assuming the annotator's error distance only depends on the object area, we use the findings of our error analysis on PASCAL VOC 2007 (Figure 5.4) to generate realistic noisy simulated clicks for COCO.

Our simulated one-click supervision approach achieves double the performance of reference MIL, reaching 51.8% CorLoc and 18.3% mAP (yellow dot in Figure 5.11). Our simulated two-click supervision approach goes even beyond that, with 58.6% CorLoc and 19.3% mAP (orange dot in Figure 5.11). Assuming the same annotation time per click as in PASCAL VOC 2007, the total annotation time for one-click is 125 hours.

Full supervision. Training with full supervision requires 2,343 hours of annotation time and leads to 24.0% mAP.

Human verification [Papadopoulos et al., 2016]. In Chapter 4 of the thesis we did not collect real human verification responses on COCO. Thus, we simulate here the human verification responses by sampling them according to the error distribution of expert humans on PASCAL VOC 2007 (see Figure 4.10). This creates a realistic simulation. The CorLoc and mAP of this scheme can be seen in Figure 5.11 (blue lines). Our two-click supervision approach reaches about the same CorLoc as our simulated human verification scheme of Chapter 4 (58.3%) and it performs a bit better in terms of mAP (19.3% vs 18.8%). Importantly, it takes about 3.5× less total annotation time. From another perspective, given the same annotation time (250 hours), our two-click supervision approach outperforms the human verification one by +16% CorLoc and +4% mAP. Hence, on difficult datasets with small objects our proposed center-click

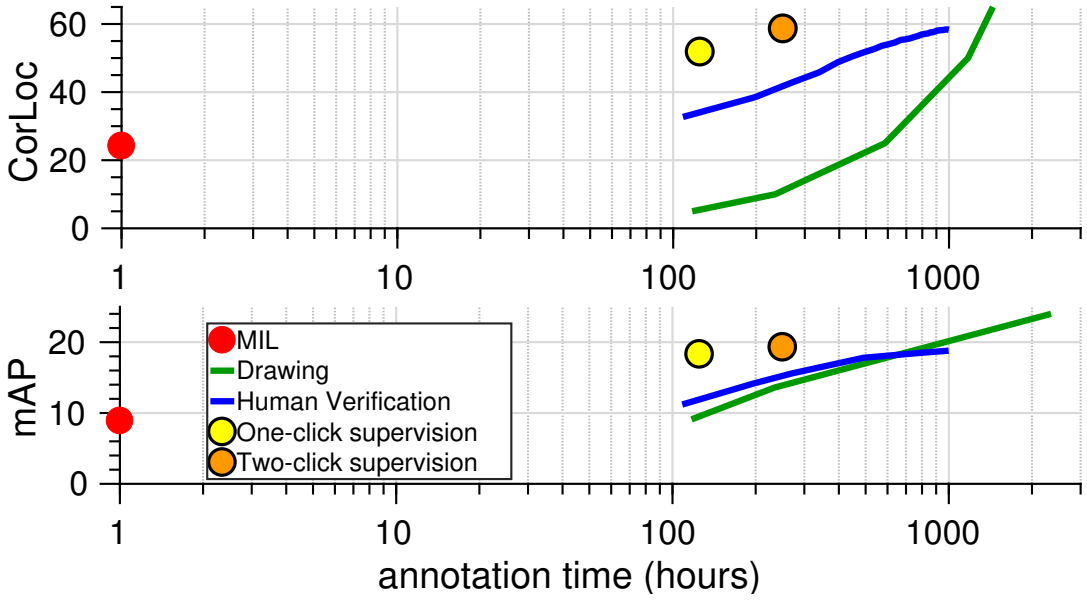


Figure 5.11: **Evaluation on MSCOCO.** CorLoc and mAP performance against human annotation time in hours(log-scale).

annotation scheme has an edge, as the efficiency of our human verification scheme of Chapter 4 degrades, while the benefits of click supervision remain. This happens probably because our initial base model (reference MIL) is weak and performs poorly on COCO. In the beginning of the verification process a lot of boxes are judged as wrong. This results in training weaker object detectors than we had on PASCAL VOC that leads again in wrong detections in the following iterations. In Section 7.2, we discuss how one can improve the performance of our models on difficult datasets.

5.6 Center clicking with an eye tracker

In this chapter, we proposed annotating objects by asking the annotators to simply (mouse-)click on the object center. In this section, we perform the same annotation task using a portable eye tracker and ask the annotators to simply *look* at the object center instead of mouse-clicking on it. The annotators are asked to press a button to signal that they found the center. The position of their last eye fixation before they press the button is the *center-fixation*. We use the center-fixations instead of the center-clicks in our method of Section 5.4 to improve the standard WSOL framework.

Intuitively, obtaining center-fixations will be faster than center-clicking as the annotators will not waste time to move the mouse on the object center. However, one

should expect that center-fixations will not be as accurate as the center clicks.

In Section 5.6.1, we describe how we collected the center-fixations with a portable eye tracker. In Section 5.6.2, we show results on incorporating center-fixations into WSOL and we compare their performance to the center-clicking scheme.

5.6.1 Collecting center-fixations

Procedure. The procedure of collecting the center-fixations is similar to the one described in Section 5.3 for collecting the center-clicks. Our annotators are given an image and the name of the target class (e.g., *cat*). They are instructed to "imagine a perfectly tight rectangular box around the object, look at the center of this box and press a button". Note that we collect center-fixations in an in-house experiment instead of crowd-sourcing them as in Section 5.3.

Data collection started with a 12-point calibration and validation. After the calibration procedure, the annotators went through a simple training phase. During this phase, they were asked to annotate (i.e., look at the center of a target object) a batch of 15 consecutive images. After each image, they receive an immediate feedback on how well they performed. We display the actual center, the point at which they were looking and the ground-truth bounding box. The goal of this phase it to make the annotators familiar with the equipment (eye tracker), the task and the interface.

After this training stage, the annotators proceed to the main annotation phase. For increased efficiency, they were shown images from a single object class at a time (for more details see Section 4.5.2). Re-calibration of the eye-tracker was performed every 15 images. This helped in minimizing calibration drift.

Apparatus. The experiment was conducted in a sound-attenuated room; participants were seated 30 cm from the screen of a 13" Macbook pro (mid 2012) while their eye movements were recorded using an Eye Tribe, which sampled both eyes at the rate of 60 Hz. A chin rest was used to minimize participants head movements and improve recording accuracy. Button press was recorded using a Logitech keyboard. Note that in contrast to the very expensive Eyelink 2000 eye tracker used for collecting the eye tracking data in Chapter 3, here we use a very cheap ($100) and portable eye tracker that can simply be attached on a tablet or a laptop.

Data collection. We collected center-fixation annotations for 5 classes (bus, cow, din-

ingtable, horse, tvmonitor) of PASCAL VOC 2007 [Everingham et al., 2007], a total of 1,070 images. Each image was annotated by two different annotators.

Annotation time. For this subset of PASCAL VOC 2007, the mean response time per image was 1.96s for center-clicks. For center-fixations we measured 1.18s per image, which is a speed-up of 1.7×. This validates our hypothesis that obtaining center-fixations is faster than center-clicking as the annotators do not waste any time to move the mouse and click on the object center. The task of estimating the center of the object and look at it is extremely fast, as we know that the visual search time to find the object is about 1s (see findings of Chapter 3).

Error analysis of center-fixations. We evaluate here the accuracy of the collected fixations by measuring their distance from the true centers of the ground-truth object bounding boxes. This error distance is 36.2 pixels on average with a median of 25.6 pixels. As a reference, the mean error distance of center-clicking on the same set of images was 19.2 pixels. We observe that center-fixations are much more noisy than center clicks. This difference in accuracy is due to a lot of factors: there is definitely a measurement error of fixations due to bad calibration; the mouse-pointer of the clicking task helps the annotators to be more accurate; the annotator training also plays an important role (note that the training stage described here is simpler than the one in Section 5.3.

5.6.2 Incorporating center-fixations into WSOL

In this section, we incorporate the collected center-fixations into a WSOL framework and we compare the results to the corresponding models that use center-clicking (Chapter 5).

Data and Evaluation. We perform experiments on 5 classes of the PASCAL VOC 2007 [Everingham et al., 2007] dataset,resulting in 1070 trainval images. We quantify localization performance on this subset with Correct Localization (CorLoc).

Results. For simplicity, we use the center-fixations into our one-click and two-click supervision models (Section 5.4) in one additional re-localization iteration at the end of the reference MIL (as opposed to using it in every iteration). For the center-fixation models, we use the values of hyper-parameters learned for our center-click annotations (Section 5.4.4).

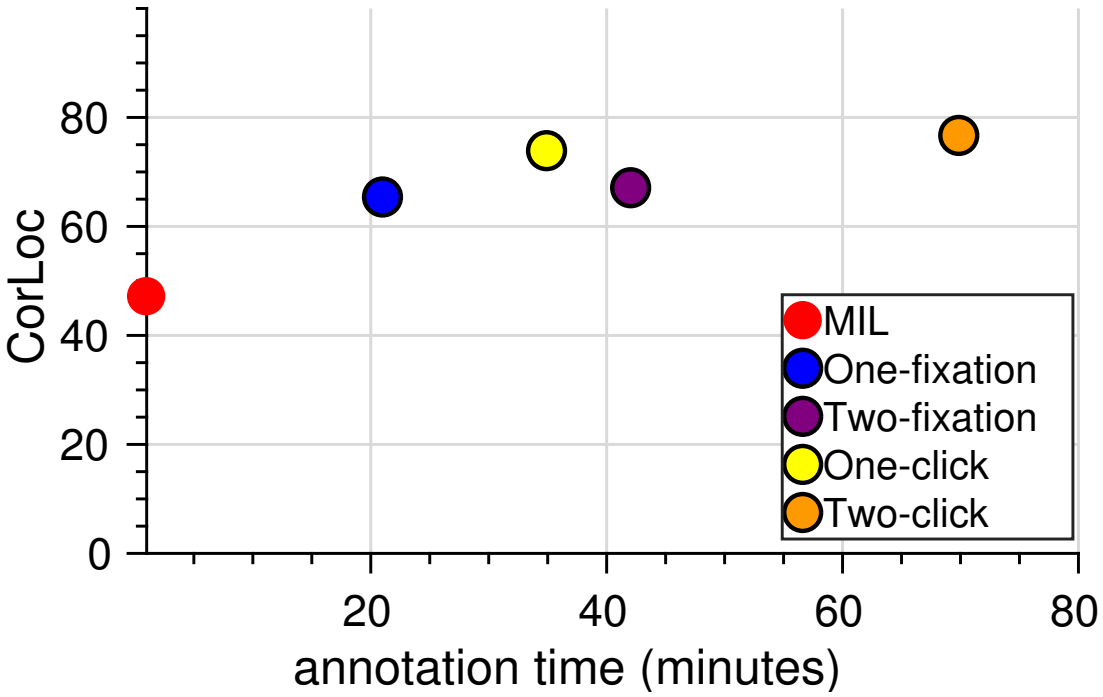


Figure 5.12: *CorLoc performance against human annotation time in minutes. We compare the one-fixation (blue dot) and two-fixation (purple dot) supervision model to the reference MIL (red dot), the one-click (yellow dot) and the two-click (orange dot) supervision model*

The CorLoc performance of all models against human annotation time in minutes is shown in Figure 5.12. The reference MIL yields 47.1% CorLoc. The one-fixation supervision model leads to 65.3%, a +18.2 improvement with only 21 minutes of extra annotation time. The two-fixation supervision doubles the annotation time (42 minutes) and reaches 67.0%. The one-click supervision requires 35 minutes of click annotation and leads to 73.8%, while the two-click supervision model leads to 76.6% with 69 minutes of annotation time.

Conclusions. In this section, we proposed annotating objects by asking the annotators to simply look at the object center and press a button. We tracked their eye movements with a very cheap and portable eye-tracker while they performed this task. Our model that incorporates center-fixations on a WSOL framework significantly improves the CorLoc performance by +18.2% with one fixation and by +19.9% with two fixations with only a modest extra annotation cost. This demonstrates our point in Section 3.5.1, that eye tracking data collected during a visual search task can be incorporated into a WSOL framework to further improve object localizations.

The task was performed very efficiently by human observers (1.2s per object per

annotator). In fact, it is 1.7× faster than the center (mouse-)clicking task. However, the accuracy of the center fixations is lower compared to the center clicks and this is why our models with mouse clicks perform better than those incorporating fixations.

We consider these preliminary results of center-fixations quite promising. Training the annotators properly for this task (i.e., similar training with center clicking based on a qualification test) can further improve the accuracy of the annotations. Also, learning the hyper-parameters of the model (Section 5.4.4) on these noisy fixations rather than on center-clicks can significantly improve the performance of the fixation supervision model.

Acknowledgments. We are grateful to Prateek Bawa for collecting the eye tracking data of this section during his undergraduate final-year project [Bawa, 2017].

5.7 Conclusions

We proposed center-click annotation as a way of training object class detectors and showed that crowd-sourced annotators can perform this task accurately and fast (1.9s per object). In extensive experiments on PASCAL VOC and MS COCO we have shown that our center-click scheme dramatically improves over weakly supervised learning of object detectors, at a modest additional annotation cost. Moreover, we have shown that it reduces total annotation time by 9×-18× compared to manually drawing bounding boxes, while still delivering high-quality detectors. Finally, we have shown that our center-click annotation scheme compares favorably against our human verification scheme (Chapter 4).

We hypothesize that our center-clicking framework is particularly well-suited for datasets that are harder than PASCAL, as the performance of baseline detectors will degrade, but the benefit of one-click supervision will remain. In this Chapter, we presented initial evidence for this claim using simulations on the challenging COCO dataset.

Chapter 6

Extreme clicking for efficient object annotation

Contents

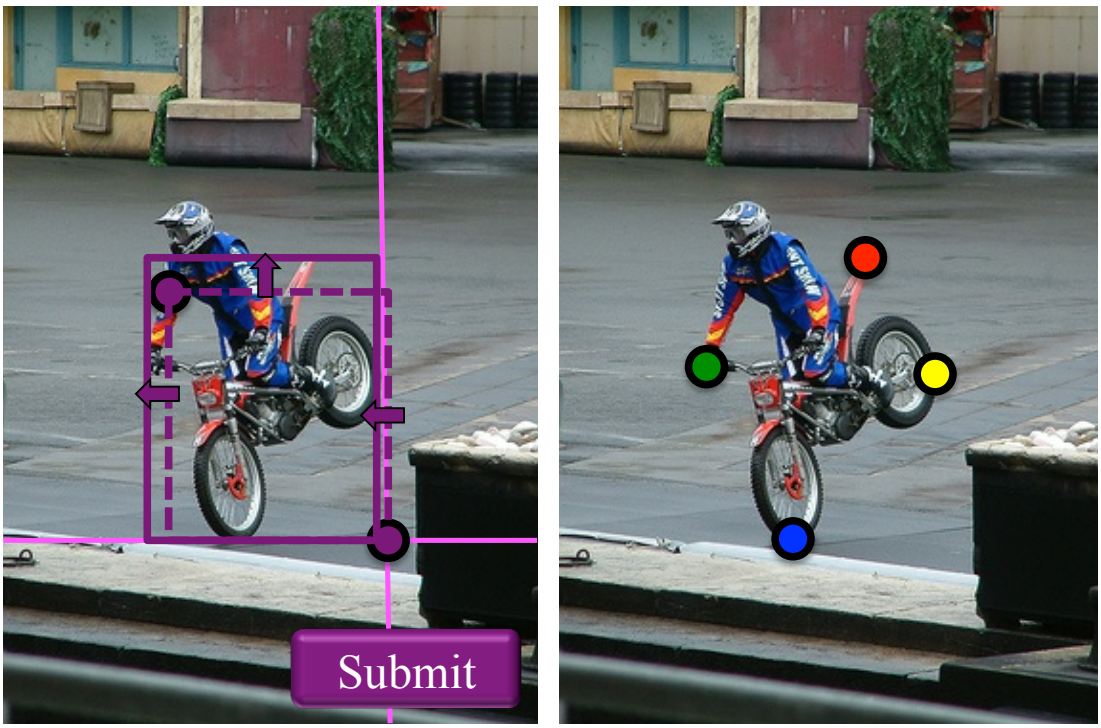


Figure 6.1: **Annotating an instance of motorbike:** (a) The conventional way of drawing a bounding box. (b) Our proposed extreme clicking scheme.

6.1 Introduction

Drawing the bounding boxes traditionally used for object detection is very expensive. Even the efficient protocol of Su et al. [2012] requires 35s to annotate one box (more details in Sec. 6.2). Why does it take so long to draw a bounding box? Fig 6.1a shows the typical process [CrowdFlower, 2016; Everingham et al., 2010; Jain and Grauman, 2013; Russakovsky et al., 2015b; Spare5/MightyAI, 2017; Su et al., 2012]. First the annotator clicks on a corner of an imaginary rectangle tightly enclosing the object (say the bottom-right corner). This is challenging, as these corners are typically not on the object. Hence the annotator needs to find the relevant extreme points of the object (the bottom point and the rightmost point) and adjust the x- and y-coordinates of the corner to match them. After this, the annotator clicks and drags the mouse to the diagonally opposite corner. This involves the same process of x- and y-adjustment, but now based on a visible rectangle. After the rectangle is adjusted, the annotator clicks again. He/she can make further adjustments by clicking on the sides of the rectangle and dragging them until the box is tight on the object. Finally, the annotator clicks a "submit" button.

From a cognitive perspective, the above process is suboptimal. The three steps (clicking on the first corner, dragging to the second corner, adjusting the sides) effectively constitute three distinct tasks. Each task requires attention to different parts of the object and using the mouse differently. In effect, the annotator is constantly task-switching, a process that is cognitively demanding and is correlated with increased response times and errors rates [Monsell, 2003; Rubinstein et al., 2001]. Furthermore, the process involves a substantial amount of mental imagery: the rectangle to be drawn is imaginary, and so are the corner points. Mental imagery also has a cognitive cost, e.g. in mental rotation experiments, response time is proportional to rotation angle [Kosslyn et al., 1995; Shepard and Metzler, 1971].

In this chapter we propose an annotation scheme which avoids task switching and mental imagery, resulting in greatly improved efficiency. We call our scheme *extreme clicking*: we ask the annotator to click on four extreme points of the object, i.e. points belonging to the top, bottom, left-most, and right-most parts of the object (Fig 6.1b). This has several advantages: (1) Extreme points are not imaginary, but are well-defined physical points on the object, which makes them easy to locate. (2) No rectangle is involved, neither real nor imaginary. This further reduces mental imagery, and avoids the need for detailed instructions defining the notion of a bounding box. (3) Only a single task is performed by the annotator thus avoiding task switching. (4) No separate box adjustment step is required. (5) No "submit" button is necessary; annotation terminates after four clicks.

Additionally, extreme clicking provides *more information* than just box coordinates: we get four points on the actual object boundary. We demonstrate how to incorporate them into GrabCut [Rother et al., 2004], to deliver more accurate segmentations than when initializing it from bounding boxes [Rother et al., 2004]. In particular, GrabCut relies heavily on the initialization of the object appearance model (e.g. [Kuettel and Ferrari, 2012; Rother et al., 2004; Wang and Cohen, 2005]) and on which pixels are clamped to be object/background. When using just a bounding box, the object appearance model is initialized from all pixels within the box (e.g. [Ferrari et al., 2008; Kuettel and Ferrari, 2012; Rother et al., 2004]). Moreover, it typically helps to clamp a smaller central region to be object [Ferrari et al., 2008]. Instead, we first expand our four object boundary points to an estimate of the whole contour of the object. We use this estimate to initialize the GrabCut object appearance model. Furthermore, we skeletonize the estimate and clamp the resulting pixels to be object.

We perform extensive experiments on PASCAL VOC 2007 and 2012 using crowd-

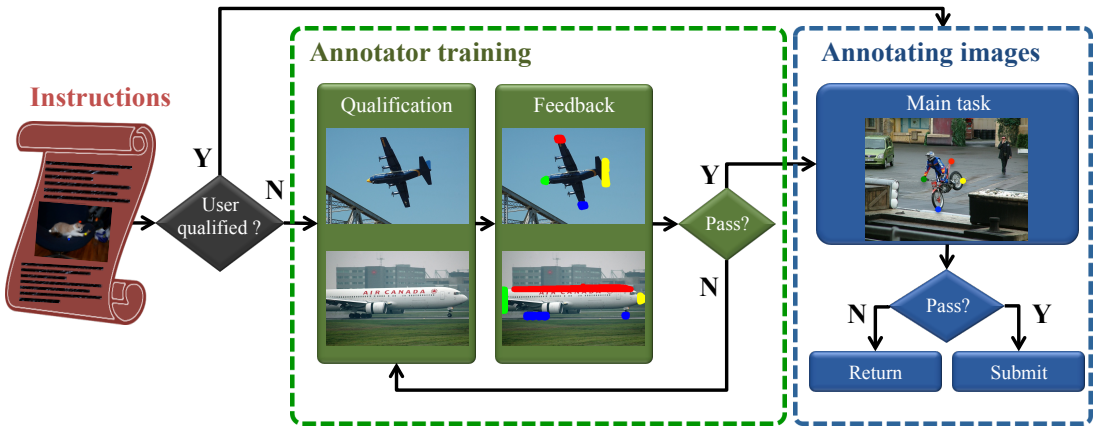


Figure 6.2: ***The workflow of our crowd-sourcing protocol for collecting extreme click annotations on images.*** *The annotators read a set of instructions and then go through an interactive training stage that consists of a qualification test at the end of which they receive a detailed feedback on how well they performed. Annotators who successfully pass the test can proceed to the annotation stage. In case of failure, they are allowed to repeat the test as many times as they want until they succeed.*

sourced annotations which demonstrate: (1) extreme clicking only takes 7s seconds per box, 5× faster than the traditional way of drawing boxes [Su et al., 2012]; (2) extreme clicking leads to high-quality boxes on a par with the original ground-truth boxes drawn the traditional way; (3) detectors trained on boxes generated using extreme clicking perform as well as those trained on the original ground-truth; (4) incorporating extreme points into GrabCut [Rother et al., 2004] improve object segmentations by 2%-4% mIoU over initializing it from bounding boxes; (5) semantic segmentations models trained on segmentations derived from extreme clicking outperform those trained on segmentations generated from bounding boxes by 2.6% mIoU.

6.2 Related work

More details about related work for drawing a bounding box can be found in Section 2.1.6. More details about WSOL or other ways to reduce annotation effort for training object class detectors can be found in Sections 2.2 and 2.3. We focus here on object segmentation work, which is not covered in previous parts of the thesis.

Object segmentations are significantly more expensive to obtain than bounding boxes. The creators of the SBD dataset [Hariharan et al., 2011] merged five annotations per instance, resulting in a total time of 315s per instance. For COCO [Lin

et al., 2014], 79s per instance were required for drawing object polygons, excluding verifying correctness and possibly redrawing. To reduce annotation time many interactive segmentation techniques have been proposed, which require the user to input either a bounding box around the object [Lempitsky et al., 2009; Rother et al., 2004; Wu et al., 2014], or scribbles [Bai and Sapiro, 2009; Duchenne et al., 2008; Freedman and Zhang, 2005; Grady, 2006; Gulshan et al., 2010; Lempitsky et al., 2009; Price et al., 2010; Veksler, 2008; Vicente et al., 2008; Yang et al., 2010], or clicks [Jain and Grauman, 2016a; Wang et al., 2014b]. Most of this work is based on the seminal GrabCut algorithm [Rother et al., 2004], which iteratively alternates between estimating appearance models (typically Gaussian Mixture Models [Blake et al., 2004]) and refining the segmentation using graph cuts [Boykov and Kolmogorov, 2004]. The user input is typically used to initialize the appearance model and to clamp some pixels to background. In this chapter, we incorporate extreme clicks into GrabCut [Rother et al., 2004], improving the appearance model initialization and automatically selecting good seed pixels to clamp to object.

6.3 Collecting extreme clicks

In this section, we describe our crowd-sourcing framework for collecting extreme click annotations (Figure 6.2). Annotators read a simple set of instructions (Section 6.3.1) and then go through an interactive training stage (Section 6.3.2). Those who successfully pass the training stage can proceed to the annotation stage (Section 6.3.3).

6.3.1 Instructions

The annotators are given an image and the name of a target object class. They are instructed to click on four extreme points (top, bottom, left-most, right-most) on the visible part of any object of this class. They can click the points in any order. In order to let annotators know approximately how long the task will take, we suggest a total time of 10s for all four clicks. This is an upper bound on the expected annotation time that we estimated from a small pilot study.

Note that our instructions are extremely simple, much simpler than those necessary to explain how to draw a bounding box in the traditional way (e.g. [Russakovsky et al., 2015b; Su et al., 2012]). They are also simpler than instructions required for verifying whether a displayed bounding box is correct [Russakovsky et al., 2015b; Su et al.,

2012]. That requires the annotator to imagine a perfect box on the object, and to mentally compare it to the displayed one (Section 4.5.2).

6.3.2 Annotator training

After reading the instructions, the annotators go through the training stage. They have to complete a qualification test, at the end of which they receive detailed feedback on how well they performed. Annotators who successfully pass this test can proceed to the annotation stage. In case of failure, annotators can repeat the test until they succeed.

Qualification test. A qualification test is a good mechanism for enhancing the quality of crowd-sourcing data and for filtering out bad annotators and spammers [Andriluka et al., 2014; Endres et al., 2010; Johnson and Everingham, 2011; Krause et al., 2013]. Some annotators do not pay attention to the instructions or do not even read them. Qualification tests have been successfully used to collect image labels, object bounding boxes, and segmentations for some of the most popular datasets (e.g., COCO [Lin et al., 2014] and Imagenet [Russakovsky et al., 2015a; Su et al., 2012]).

The qualification test is designed to mimic our main task of clicking on the extreme points of objects. We show the annotator a sequence of 5 different images with the same object class and ask them to carry out the extreme clicking task.

Feedback. The qualification test uses a small pool of images with ground-truth segmentation masks for the objects, which we employ to automatically validate the annotator's clicks and to provide feedback (Figure 6.2, middle part). We take a small set of qualification images from a different dataset than the one that we annotate.

In the following, we explain the validation procedure for the top click (the other three cases are analogous). We ask the annotator to click on a top point on the object, but this point is not necessarily uniquely defined. Depending on the object shape, there may be multiple points that are equivalent, up to some tolerance margin (e.g. the top of the dog's head in Figure 6.3, top row). Clearly, clicking on any of these points is correct. The area in which we accept the annotator's click is derived from the segmentation mask. First, we find the pixels with the highest y-coordinate in it (there might be multiple such pixels). Then, we select all pixels in the mask with y-coordinates within 10 pixels of any of these top pixels (red area in Figure 6.3, middle column). Finally, we also include in the accepted area all image pixels within 10 pixels of any of the selected pixels in the segmentation mask (Figure 6.3, right column). Thus

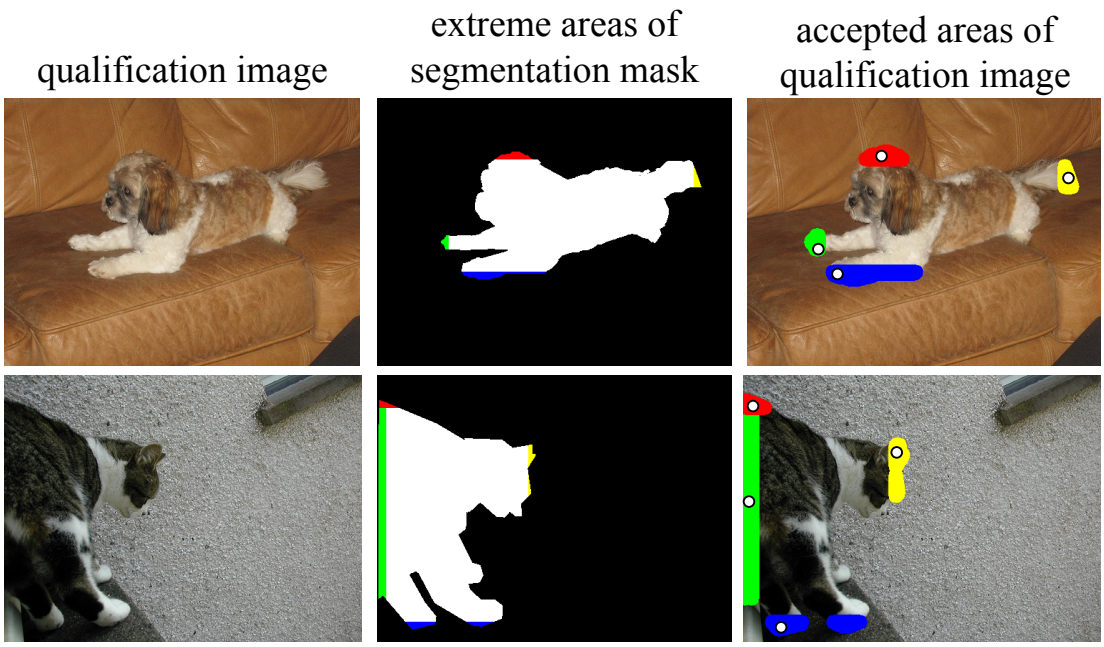


Figure 6.3: ***Qualification test.*** *(Left) Qualification test examples of the dog and cat class. (Middle) The figure-ground segmentation masks we use to evaluate annotators' extreme clicks during the training stage. The pixels of the four extreme areas of the mask are marked with colors. (Right) The accepted areas for each extreme click and the click positions as we display them to the annotators as feedback.*

the accepted area includes all top pixels in the mask, plus a tolerance region around them, both inside and outside the mask.

After the annotators finish the qualification test, they receive a feedback page with all the examples they annotated. For each image, we display the annotator's four clicks, and the accepted areas for each click (Figure 6.3 right column).

Success or failure. The annotators pass the qualification test if all their clicks on all 5 qualification images are inside the accepted areas. Those that pass the test are recorded as qualified annotators and can proceed to the main annotation stage. A qualified annotator never has to retake the qualification test. In case of failure, annotators are allowed to repeat the test as many times as they want. The combination of automatically providing rich feedback and allowing annotators to repeat the test makes the training stage interactive and highly effective. Annotators that have reached the desired level of quality can be expected to keep it throughout the annotation [Hata et al., 2017].

6.3.3 Annotating images

In the annotation stage, annotators are asked to annotate small batches of 10 consecutive images. To increase annotation efficiency, the target class for all the images within a batch is the same. This means annotators do not have to re-read the class name for every image and can use their prior knowledge of the class to find it rapidly in the image [Torralba et al., 2006]. More generally, it avoids task-switching which is well-known to increase response time and decrease accuracy [Rubinstein et al., 2001; Monsell, 2003].

Quality control. Quality control is a common process when crowd-sourcing image annotations [Bearman et al., 2016; Kovashka and Grauman, 2015; Lin et al., 2014; Russakovsky et al., 2015a; Russell et al., 2008; Sorokin and Forsyth, 2008; Su et al., 2012; Vondrick et al., 2013; Welinder et al., 2010]. We control the quality of the annotation by hiding one evaluation image for which we have a ground-truth segmentation inside a 10-image batch, and monitor the annotator's accuracy on it (golden question). Annotators that fail to click inside the accepted areas on this evaluation image are not able to submit the task. We do not do any post-processing rejection of the submitting data.

6.4 Object segmentation from extreme clicks

Extreme clicking results not only in high-quality bounding box annotations, but also in four accurate object boundary points. In this section we explain how we use these boundary points to improve the creation of segmentation masks from bounding boxes.

We cast the problem of segmenting an object instance in image I as a pixel labeling problem. Each pixel $p \in I$ should be labeled as either object ($l_p = 1$) or background ($l_p = 0$). A labeling L of all pixels represents the segmented object. Similar to Rother et al. [2004], we employ a binary pairwise energy function E defined over the pixels and their labels.

$$E(L) = \sum_p U(l_p) + \sum_{p,q} V(l_p, l_q) \tag{6.1}$$

U is a unary potential that evaluates how likely a pixel p is to take label l_p according to the object and background appearance models, while the pairwise potential V encourages smoothness by penalizing neighboring pixels taking different labels.

Initial object surface estimate from extreme clicks. For GrabCut to work well,

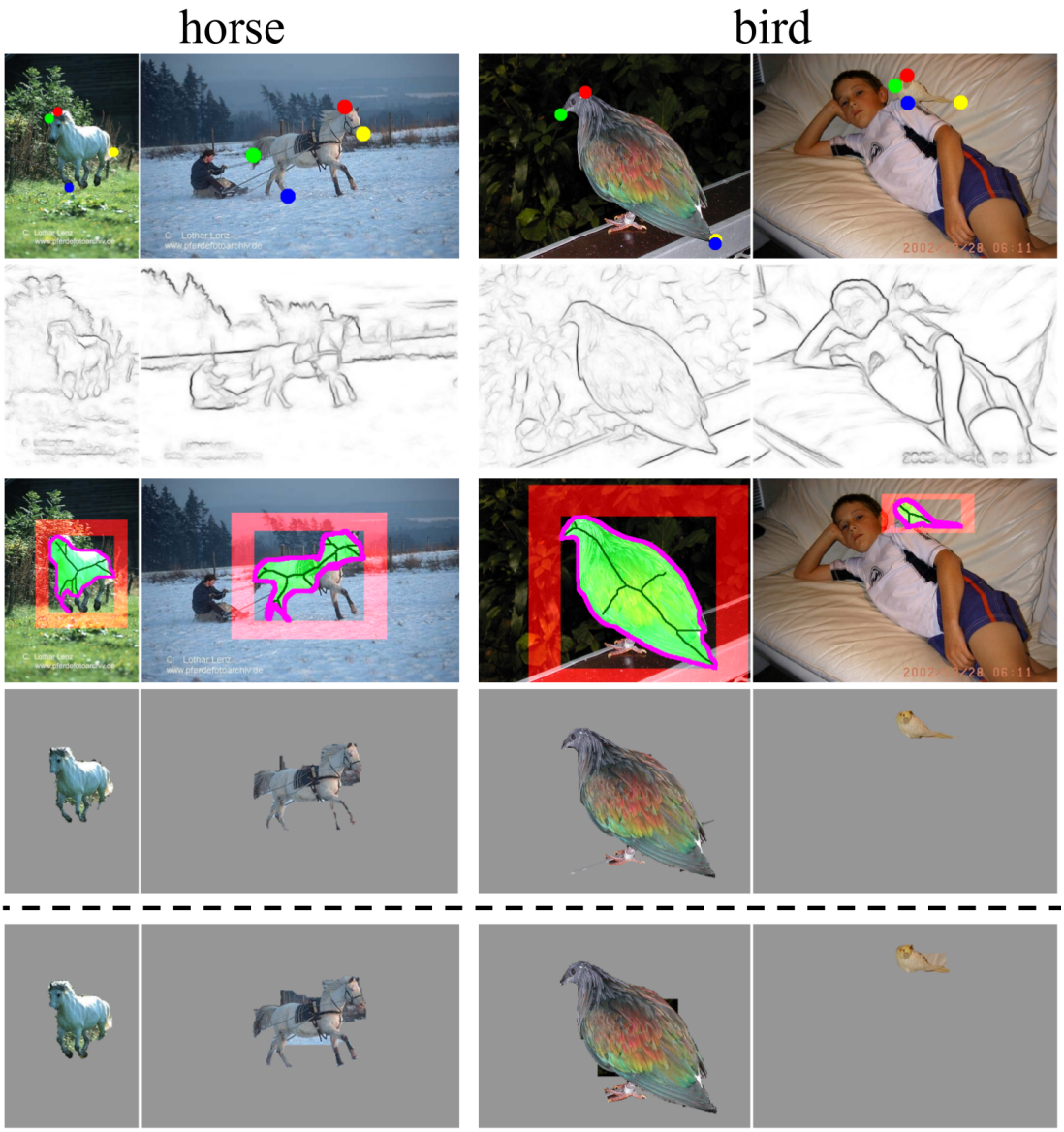


Figure 6.4: ***Visualization of input cues and output of GrabCut.*** *First row shows input with annotator's extreme clicks. Second row shows output of edge detector [Dollar and Zitnick, 2013]. Third row shows our inputs for GrabCut: the pixels used to create background appearance model (red), the pixels used to create the object appearance model (bright green), the initial boundary estimate (magenta), and the skeleton pixels which we clamp to have the object label (dark green). Fourth row shows the output of GrabCut when using our new inputs, while the last row shows the output when using only a bounding box.*

it is important to have a good initial estimate of the object surface to initialize the appearance model. Additionally, it helps to clamp certain pixels to object [Kuettel and Ferrari, 2012]. We show how the four collected object boundary points can be exploited to do both.

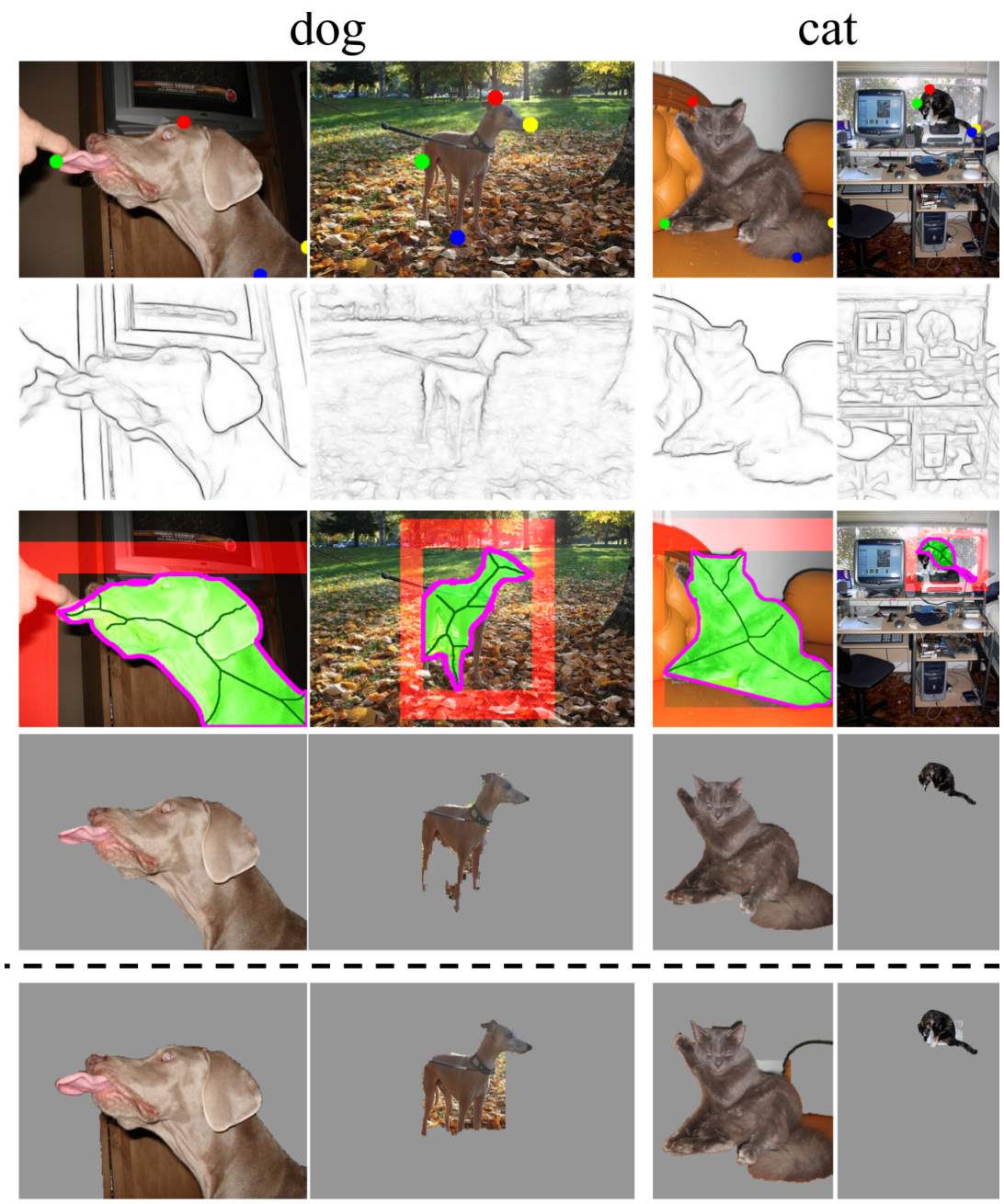


Figure 6.5: *Extra examples of input cues and output of GrabCut (For more details see caption of Figure 6.4.*

In particular, for each pair of consecutive extreme clicks (e.g. leftmost-to-top, or top-to-rightmost) we find the path connecting them which is most likely to belong to the object boundary. For this purpose we first apply a strong edge detector [Dollar and Zitnick, 2013] to obtain a boundary probability $e_p \in [0,1]$ for every pixel p of the image (second row of Figures 6.4, 6.5). We then define the best boundary path between two consecutive extreme clicks as the shortest path whose minimum edge-response is

the highest (third row of Figures 6.4 , 6.5, magenta). In practice, we compute this path by thresholding the output of the edge detector with a high value and iteratively reduce it until a continuous line that connects the two extreme clicks exists. We found this objective function to work better than others, such as minimizing $\sum_p(1-e_p)$ for pixels p on the path. The resulting object boundary paths yield an initial estimate of the object outlines.

We use the surface within the boundary estimates (shown in green in the third row of Figures 6.4, 6.5) to initialize the object appearance model used for U in Equation (6.1). Furthermore, from this surface we obtain a skeleton using standard morphology (shown in dark green in third row of Figures 6.4, 6.5). This skeleton is very likely to be object, so we clamp its pixel-labels to be object ($l_s = 1$ for all pixels s on the skeleton).

Appearance model. As in classic GrabCut [Rother et al., 2004], the appearance model consists of two GMMs, one for the object (used when $l_p = 1$) and one for the background (used when $l_p = 0$). Each GMM has five components, where each is a full-covariance Gaussian over the RGB color space.

Traditional interactive segmentation techniques [Lempitsky et al., 2009; Rother et al., 2004; Wu et al., 2014] start from a manually drawn bounding box and estimate the initial appearance models from all pixels inside the box (object model) and all pixels outside it (background model). However, this may be suboptimal: since we are trying to segment the object within the box, intuitively only the immediate background is relevant, not the whole image. Indeed, we improved results by using a small rectangular ring around the bounding box for initializing the background model (see third row Figures 6.4, 6.5 in red). Furthermore, not all pixels within the box belong to the object. But given only a bounding box as input, the best is to still use the whole box to initialize the object model. Therefore, in our baseline GrabCut implementation, the background model is initialized from the immediate background and the object model is initialized from all pixels within the box.

However, because we have extreme clicks we can do better. We use them to obtain an initial object surface estimate (described above) from which we initialize the object appearance model. Fig. 6.6 illustrates how this improves the unary potentials U resulting from the appearance models.

Clamping pixels. GrabCut sometimes decides to label all pixels either as object or background. To prevent this, one can clamp some pixels to a certain label. For the

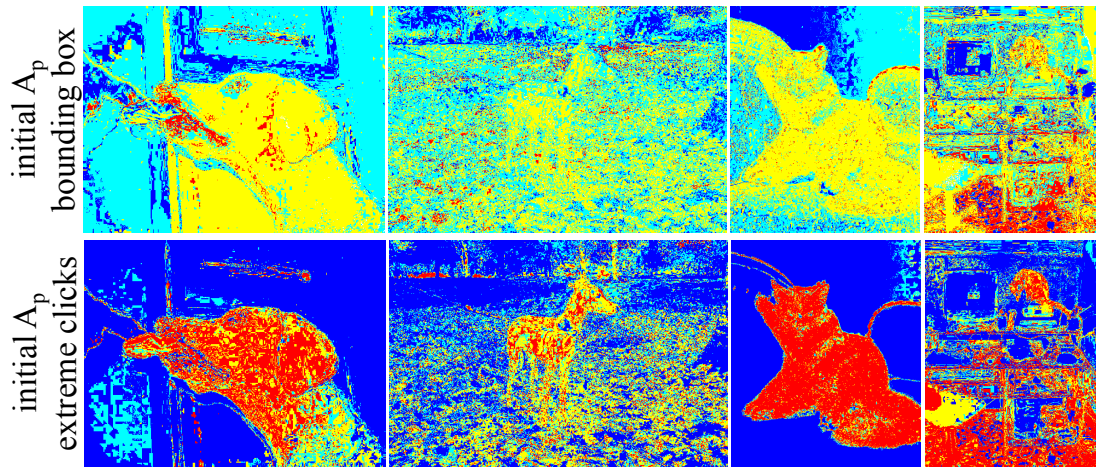


Figure 6.6: *Posterior probability of pixels belonging to object. For both rows the background appearance model is created by using an area outside the initial box (see Figure 6.5). In the first row the object model is created using the area inside the initial box. In the second row the object model is created from the object surface estimated using extreme clicks (Figure 6.5, third row in light-green). Predictions from the appearance model using extreme clicks are visibly better.*

background, all pixels outside the bounding box are typically clamped to background. For the object, one possible approach is to clamp a small area in the center of the box [Ferrari et al., 2008]. However, there is no guarantee that the center of the box is on the object, as many objects are not convex. Moreover, the size of the area to be clamped is not easy to set.

In this paper, we estimate the pixels to be clamped by skeletonizing the object surface estimate derived from our extreme clicks (described above). In Sec. 6.6 we show how our proposed object appearance model initialization and clamping scheme affect the final segmentation quality.

Pairwise potential V. The summation over (p,q) in (6.1) is defined on an eight-connected pixel grid. Usually, this penalty depends on the RGB difference between pixels, being smaller in regions of high contrast [Boykov and Jolly, 2001; Blake et al., 2004; Gulshan et al., 2010; Lempitsky et al., 2009; Rother et al., 2004; Veksler, 2008]. In this paper, we instead use the sum of the edge responses of the two pixels given by the edge detector [Dollar and Zitnick, 2013]. In Sec. 6.6 we evaluate both pairwise potentials and show how they affect the final segmentation.

Optimization. After the initial estimation of appearance models, we follow Rother

		Annotation quality w.r.t. GT SegBoxes			Detector performance (mAP)		Annotation time	
Dataset	Annotation approach	mIoU	IoU>0.7	IoU>0.5	AlexNet	VGG16	dataset (h)	instance (s)
PASCAL VOC 2007	Extreme clicks	88	92	98	56	66	14.3	7.0
	PASCAL GT Boxes	88	93	98	56	66	70.0	34.5
PASCAL VOC 2012	Extreme clicks	87	91	95	52	62	16.8	7.2
	PASCAL GT Boxes	87	90	96	52	62	79.8	34.5

Table 6.1: *Comparison of extreme clicking and PASCAL VOC ground-truth.*

		Annotation quality w.r.t. GT Boxes			Detector performance (mAP)		Annotation time	
Dataset	Annotation approach	mIoU	IoU>0.7	IoU>0.5	AlexNet	VGG16	dataset (h)	instance (s)
PASCAL VOC 2007	Extreme clicks	88	94	97	56	66	14.3	7.0
	Human verification (Chapter 4)	58	34	76	47	54	8.9	4.4
	Center-clicking (Chapter 5)	64	45	79	49	58	7.6	3.8
	WSOL Bilen and Vedaldi [2016]	–	–	54	35	35	0	0
ILSVRC (subset)	box drawing in Russakovsky et al. [2015b]	–	71	–	–	–	–	12.3

Table 6.2: *Comparison of extreme clicking and alternative fast annotation approaches.*

et al. [2004] and alternate between finding the optimal segmentation L given the appearance models, and updating the appearance models given the segmentation. The first step is solved globally optimally by minimizing (6.1) using graph-cuts [Boykov and Kolmogorov, 2004], as our pairwise potentials are submodular. The second step simply fits GMMs to labeled pixels.

6.5 Extreme Clicking Results

We implement our annotation scheme on Amazon Mechanical Turk (AMT) and collect extreme click annotations for both the trainval set of PASCAL VOC 2007 [Everingham et al., 2007] (5011 images) and the training set of PASCAL VOC 2012 [Everingham et al., 2012] (5717 images), which contain 20 object categories. For every image we annotate a single instance per class (if present in the image), which enables direct comparison to other methods described below. We compare methods both in terms of efficiency and quality.

Compared methods. Our main comparisons are to the existing ground-truth bounding

boxes of PASCAL VOC. As discussed in Sec. 2.1.6, we use 34.5s as the reference time necessary to produce one such high quality bounding box by drawing it the traditional way [Su et al., 2012].

At the other extreme, it is possible to obtain lower quality bounding boxes automatically at zero extra costs by using weakly supervised methods, which only input image-level labels. We compare to the recent method of Bilen and Vedaldi [2016].

We also compare to three methods which strike a trade-off between accuracy and efficiency: Russakovsky et al. [2015b], our proposed human verification scheme (Chapter 4) and our proposed center-clicking scheme (Chapter 5). In Russakovsky et al. [2015b], manual box drawing is part of a complex computer-assisted annotation system. Importantly, Russakovsky et al. [2015b] report both annotation time and quality, enabling proper comparisons. In Chapter 4 we proposed an annotation scheme that only requires annotators to verify boxes automatically generated by a learning algorithm [Papadopoulos et al., 2016], while in Chapter 5 we propose another efficient annotation scheme that only requires annotators to click on the center of the objects [Papadopoulos et al., 2017].

Evaluation measures. For evaluating efficiency we report time measurements, both in terms of annotating the whole dataset and per instance.

We evaluate the quality of bounding boxes with respect to the PASCAL VOC ground-truth. We do this with respect to the ground-truth bounding boxes (*GT Boxes*), but also with respect to bounding boxes which we fit to the ground-truth *segmentations* (*GT SegBoxes*). We quantify quality by intersection-over-union (IoU) [Everingham et al., 2010], where we measure the percentage of bounding boxes we annotated per object class with IoU greater than 0.5 and 0.7, and then take the mean over all classes (IoU$>$0.5, IoU$>$0.7). In addition, we calculate the average IoU for all instances of a class and take the mean over all classes (mIoU).

As an additional measure of accuracy we measure detector performance using Fast R-CNN [Girshick, 2015], trained either on our extreme click boxes or on the PASCAL GT Boxes.

6.5.1 Results on quality and efficiency

PASCAL ground-truth boxes vs. extreme clicks. Table 6.1 reports the results. Having two sets of ground-truth boxes enables us to measure the agreement among the

expert annotators that created PASCAL. Comparing GT Boxes and GT SegBoxes reveals this agreement to be at 88% mIoU on VOC 2007. Moreover, 93% of all GT Boxes have IoU > 0.7 with their corresponding GT SegBox. This shows that the ground-truth annotations are highly consistent, and these metrics represent the quality of the ground-truth itself. Similar findings apply to VOC 2012.

Interestingly, the boxes derived from our extreme clicks achieve equally high metrics, when compared to the PASCAL GT SegBoxes. Therefore our extreme click annotations yield boxes with a quality within the agreement among expert-annotators using the traditional way of drawing. To get a better feeling for such quality, if we perturb each of the four coordinates of the GT Boxes by 4 pixels, the resulting boxes also have 88% mIoU with the unperturbed annotations. Qualitative examples are shown in the appendix A.

To further demonstrate the quality of extreme clicking, we train Fast R-CNN [Girshick, 2015] using either PASCAL GT Boxes or extreme click boxes. We train on PASCAL VOC 2007s trainval set and test on its test set, then we train on VOC 2012s train and test on its val set. We experiment using AlexNet [Krizhevsky et al., 2012] and VGG16 [Simonyan and Zisserman, 2015]. Performance when training from GT Boxes or from our boxes is identical on both datasets and using both base networks.

Annotation efficiency. In terms of annotation efficiency, extreme clicks are 5× cheaper: 7.0s instead of 34.5s. This demonstrates that extreme clicking costs only a fraction of the annotation time of the widely used box-drawing protocol [CrowdFlower, 2016; Everingham et al., 2010; Russakovsky et al., 2015b; Spare5/MightyAI, 2017; Su et al., 2012], without any compromise on quality.

Human verification (Chapter 4) vs. extreme clicks. Table 6.2 compares extreme clicks to our human verification scheme of Chapter 4 on PASCAL VOC 2007. For a fair comparison, we compare to using human verifications by AMT annotators. While verification is 1.6× faster, our bounding boxes here are much more accurate (97% correct at IoU>0.5, compared to 76% for human verification). Additionally, detector performance at test time is 9%-12% mAP higher for extreme clicking.

Center clicks (Chapter 5) vs. extreme clicks. Table 6.2 also compares extreme clicks to our center clicks of Chapter 4 on PASCAL VOC 2007. As in Chapter 5 we use only two center-clicks as opposed to the four extreme clicks here. Center clicking is 1.9× faster than extreme clicking. However, extreme clicking bounding boxes are much

more accurate (97% correct at IoU>0.5, compared to 79% for center clicking) leading to higher detector performance at test time (7%-8% mAP higher). More importantly, we expect only a small gain in performance of our center clicking scheme by increasing the number of center clicks to four. This is because the four center clicks will only allow as to estimate the object center and area even more accurately. This will not bring any truly new information.

Weak supervision vs. extreme clicks. Weakly supervised methods are extremely cheap in human supervision time. However, the recent work of Bilen and Vedaldi [2016] reports 35% mAP using VGG16, which is only about half the result brought by extreme clicking (66% mAP, Table 6.2).

Box drawing [Russakovsky et al., 2015b] vs. extreme clicks. Finally, we compare to Russakovsky et al. [2015b] in Table 6.2. This is an approximate comparison as measurements of their box-drawing component are done on an unspecified subset of ILSVRC 2014. However, as ILSVRC and PASCAL VOC are comparable in both quality of annotations and difficulty of the dataset [Russakovsky et al., 2015a], this comparison is representative. In Russakovsky et al. [2015b] they report 12.3s for drawing a bounding box, where 71% of the drawn boxes have an IoU>0.7 with the ground-truth box. This suggests that bounding boxes can be drawn faster than reported in Su et al. [2012] but this comes with a significant drop in quality. In contrast, extreme clicking costs 7s per box and 91%-94% of those boxes have IoU>0.7. Hence our protocol to annotate bounding boxes is both faster and more accurate.

6.5.2 Additional analysis

We now report several additional experiments and measurements providing additional insights.

Oracle best window proposals [Zitnick and Dollár, 2014]. Many weakly supervised methods (e.g. [Siva and Xiang, 2011; Deselaers et al., 2012; Russakovsky et al., 2012; Cinbis et al., 2014; Bilen and Vedaldi, 2016]) are based on object proposals such as Alexe et al. [2010]; Uijlings et al. [2013]; Zitnick and Dollár [2014]. Therefore, it is interesting to discover what the theoretical upper bound of quality is than can be reached by using proposals. To do this, we use EdgeBoxes [Zitnick and Dollár, 2014] and take for each object the proposal which has the highest overlap with the

Qualification test	Quality control	mIoU	IoU>0.7
		75.4	68.0
✓		85.7	91.0
✓	✓	87.1	92.5

Table 6.3: Influence of the qualification test and quality control on the accuracy of extreme click annotations (on 200 images from PASCAL VOC 2007).

ground truth bounding box (among 2000 proposals per image). On PASCAL VOC 2007 we measure 82% mIoU and 91% of them are correct at IoU>0.7. Extreme clicking yields 88% mIoU and 94% correct boxes at IoU>0.7. Results also transfer to object detection performance: Fast R-CNN using VGG16 achieves 62% mAP for the best proposal while extreme clicks yield 66% mAP. We conclude that weakly supervised methods can never select object proposals which are better than boxes obtained through extreme clicking.

Per-click response-time. We examine the mean response time per click during extreme clicking. Interestingly, the first click on an object takes about 2.5s, while subsequent clicks take about 1.5s. This is because the annotator needs to find the object in the image before they can make the first click. Interestingly, 1s visual search time per object is again consistent with our findings [Papadopoulos et al., 2014] in Chapter 3 and the findings of Ehinger et al. [2009].

Influence of qualification test and quality control. We conducted three crowdsourcing experiments on 200 trainval images of PASCAL VOC 2007 to test the influence of using a qualification test and quality control. We report the quality of the bounding boxes derived from extreme clicks in Table 6.3. Using a qualification test vastly improves annotation quality (from 75.4% to 85.7% mIoU). The quality control brings a smaller further improvement to 87.1% mIoU.

Cost. We paid the annotators \$0.15 to annotate a batch of 10 images which, based on our timings, is about \$7.7 per hour. The total cost for annotating the whole trainval set of PASCAL VOC 2007 and the training set of PASCAL VOC 2012 was \$147 and \$167, respectively.

6.6 Results on Object Segmentation

This section demonstrates that one can improve segmentation from a bounding box by using also the boundary points which we obtain from extreme clicking.

6.6.1 Results on PASCAL VOC

Datasets and Evaluation. We perform experiments on VOC 2007 and VOC 2012. The trainval set of the segmentation task of VOC 2007 consists of 422 images with ground-truth segmentation masks of 20 classes. For VOC 2012, we evaluate on the training set, using as reference ground-truth the augmented masks set by Hariharan et al. [2011] (5623 images).

To evaluate the output object segmentations, for every class we compute the intersection over union (IoU) between the predicted and ground-truth segmentation mask, and report the mean IoU over all object classes (mIoU). Some pixels in VOC 2007 are labeled as 'unknown' and are excluded from evaluation. For these experiments we use structured edge forests [Dollar and Zitnick, 2013] to predict object boundaries, which is trained on BSD500 [Arbeláez et al., 2011].

GrabCut from PASCAL VOC GT Boxes. We start with establishing our baseline by using GrabCut on the original GT Boxes of VOC (for which no boundary points are available). Since applying Rother et al. [2004] directly leads to rather poor performance on VOC 2007 (37.3% mIoU), we first optimize GrabCut on this dataset using methods discussed in Sec. 6.4. Our optimized model has the following properties: the object appearance model is initialized from all pixels within the box. The background appearance model is initialized from a small rectangular ring around the box obtained by enlarging the window by a factor $\sqrt{2}$ in all directions. A small rectangular core centered within the box whose area is a quarter of the area of the box is clamped to be object. All pixels outside the box are clamped to be background. As pairwise potential, instead of using standard RGB differences, we use the summed edge responses of Dollar and Zitnick [2013] of the corresponding pixels. All modifications together substantially improve results to 74.4% mIoU on VOC 2007. We then run GrabCut again on VOC 2012 using the exact same settings optimized for VOC 2007, obtaining 71.0% mIoU.

GrabCut from extreme clicking. Thanks to our extreme clicking annotations, we also

have object boundary points. Starting from the optimized GrabCut settings established in the previous paragraph, we make use of these boundary points to (1) initialize a better object appearance model, and (2) choose better pixels to clamp to object. As described in Sec. 6.4, we use the extreme clicks to estimate an initial contour of the object by following predicted object boundaries [Dollar and Zitnick, 2013]. We use the surface bounded by this contour estimate to initialize the appearance model. We also skeletonize this surface and clamp the resulting pixels to be object. The resulting model yields 78.1% mIoU on VOC 2007 and 72.7% on VOC 2012. This is an improvement of 3.7% (VOC 2007) and 1.7% (VOC 2012) over the strong baseline we built. Fig. 6.4 shows qualitative results comparing GrabCut segmentations starting from GT Boxes (last row) and those based on our extreme clicking annotations (second-last row).

6.6.2 Results on the GrabCut dataset

We also conducted an experiment on the Grabcut dataset [Rother et al., 2004], consisting of only 50 images. The standard evaluation measure is the error rate in terms of the percentage of mislabeled pixels. For this experiment, we simulate the extreme click annotation by using the extreme points of the ground-truth segmentation masks of the images.

When we perform GrabCut from bounding boxes, we obtain an error rate of 8%. When using additionally the boundary points from simulated extreme clicking, we obtain 5.5% error, an improvement of 2.5%. This again demonstrates that boundary points contain useful information over bounding boxes alone for this task.

For completeness, we note that the state-of-the-art method on this dataset has 3.6% error [Wu et al., 2014]. This method uses a framework of superpixels and Multiple Instance Learning to turn a bounding box into a segmentation mask. In this chapter we build on a much simpler segmentation framework (GrabCut). We believe that incorporating our extreme clicks into Wu et al. [2014] would bring further improvements.

6.6.3 Training a semantic segmentation model

We now explore training a modern deep learning system for semantic segmentation from the segmentations derived from extreme clicking. We train DeepLab [Chen et al., 2015; Papandreou et al., 2015] based on VGG-16 [Simonyan and Zisserman, 2015] on the VOC 2012 train set (5,623 images) and then we test on its val set (1,449 images). We measure performance using the standard mIoU measure (Table 6.4). We compare

	Full supervision	Segments from GT Boxes	Segments from extreme clicks
mIoU	59.9	55.8	58.4

Table 6.4: *Segmentation performance on the val set of PASCAL VOC 2012 dataset using different types of annotations.*

our approach to full supervision by training on the same images but using the ground-truth, manually drawn object segmentations (one instance per class per image, for fair comparison). We also compare to training on segmentations generated from GT Boxes.

Full supervision yields 59.9% mIoU, which is our upper bound. As a reference, training on manual segmentations for all instances in the dataset (i.e., not restrict it to one instance per class per image) yields 63.8% mIoU. This is 3.8% lower than in Papandreou et al. [2015] since they train from train+val using the extra annotations by Hariharan et al. [2011] (10.3k images).

Segments from GT Boxes result in 55.8% mIoU.

Segments from extreme clicks lead to 58.4% mIoU. This means our extreme clicking segmentations lead to a +2.6% mIoU improvement over those generated from bounding boxes. Moreover, our result is only -1.5% mIoU below the fully supervised case (given the same total number of training samples).

6.7 Conclusions

We presented an alternative to the common way of drawing bounding boxes, which involves clicking on imaginary corners of an imaginary box. Our alternative is extreme clicking: we ask annotators to click on the top, bottom, left- and right-most points of an object, which are well-defined physical points. We demonstrate that our method delivers bounding boxes that are as good as traditional drawing, while taking just 7s per annotation. To achieve this same level of quality, traditional drawing needs 34.5s [Su et al., 2012]. Hence our method cuts annotation costs by a factor 5$\times$ without any compromise on quality.

In addition, extreme clicking leads to more than just a box: we also obtain accurate object boundary points. To demonstrate their usefulness we incorporate them into GrabCut, and show that they leads to better object segmentations than when initializing

it from the bounding box alone. Finally, we have shown that semantic segmentation models trained on these segmentations perform close to those trained with manually drawn segmentations (when given the same total number of samples).

Chapter 7

Conclusions

Contents

In this thesis we proposed four efficient human annotation schemes for training object class detectors: eye-tracking, human verification, center-clicking and extreme-clicking. Our schemes reduce the human annotation cost of drawing bounding boxes while still obtaining high quality object detectors.

This chapter is organized as follows: Section 7.1 briefly summarizes the contributions of the thesis. Section 7.2 draws general conclusions by comparing our schemes. Section 7.3 gives directions for future research.

7.1 Summary of contributions

The first contribution is designing a scheme to exploit eye-tracking for training detectors. Instead of carefully marking every training image with accurate and tight bounding boxes, the annotator simply needs to perform a visual search task, i.e. looking for the object in the training images. We tracked the eye movements of annotators while they performed this task and we proposed a novel technique to derive object bounding boxes from these eye movement data. To validate our idea, we collected eye tracking data for the complete trainval set of ten object classes from PASCAL VOC 2012 [Everingham et al., 2012]. We showed that this task can be performed in a frac-

tion of the time it takes to draw a bounding box (about only one second per image). The limitations of this scheme are explained in Section 3.6. The main drawback of eye tracking is the difficulty for crowd-sourcing it, which is essential for building a very large scale dataset. Being able to accurately track eye movements with conventional web cameras could bypass this difficulty and make crowd-sourcing viable for eye tracking data [Skovsgaard et al., 2013; Xiao et al., 2015; Xu et al., 2015; Wood et al., 2015; Krafka et al., 2016; Papoutsaki et al., 2016].

The second contribution is a new efficient annotation scheme that only requires humans to verify bounding boxes produced by a learning algorithm. Our scheme introduces a human verification step to improve the re-training and re-localization steps common to most weakly supervised approaches. Extensive experiments on PASCAL VOC 2007 with both expert and crowd-sourced annotators show that our scheme produces detectors performing almost as good as those trained in a fully supervised setting, without ever drawing any bounding boxes. As the verification task is very quick, our scheme reduces the total human annotation time by a factor of 8×.

The third contribution is center-click annotation as a way of training object class detectors. We simply ask annotators to click on the center of an imaginary bounding box which tightly encloses the object instance. We then incorporate these clicks into existing MIL techniques for WSOL, to jointly localize object bounding boxes over all training images. We showed that crowd-sourced annotators can perform this task accurately and fast (1.9s per object). In extensive experiments on PASCAL VOC and MS COCO we showed that our center-click scheme dramatically improves over weakly supervised learning of object detectors, at a modest additional annotation cost. Moreover, we showed that it reduces total annotation time by 9×-18× compared to manually drawing bounding boxes, while still delivering high-quality detectors. Finally, we showed that our center-click annotation scheme compares favorably against our human verification scheme (Chapter 4).

The fourth contribution is extreme clicking: an alternative to the common way of drawing bounding boxes, which involves clicking on imaginary corners of an imaginary box. Instead, we ask annotators to click on the top, bottom, left- and right-most points of an object, which are well-defined physical points. We demonstrated that our method delivers bounding boxes that are as well as traditional drawing, while taking just 7s per box. To achieve this same level of quality, traditional drawing needs

34.5s [Su et al., 2012]. Hence our method cuts annotation costs by a factor 5× without any compromise on quality.

In addition, extreme clicking leads to more than just a box: we also obtain accurate object boundary points. To demonstrate their usefulness we incorporate them into GrabCut, and show that they lead to better object segmentations than when initializing it from the bounding box alone. Finally, we have shown that semantic segmentation models trained on these segmentations perform close to those trained with manually drawn segmentations (when given the same total number of samples).

7.2 Guidelines for choosing an annotation scheme

In this section, we compare the human verification (Chapter 4), the center-clicking (Chapter 5) and the extreme clicking scheme (Chapter 6) to each other (see Figures 7.1 and 7.2), and we draw conclusions that can be used as guidelines for choosing the most suitable annotation technique in a given situation.

To ensure a fair comparison, we evaluate all these schemes on the same dataset (i.e., PASCAL VOC 2007). For training we use the trainval set of PASCAL VOC 2007 that contains 5011 images over 20 object classes and for testing we use its test set that contains 4952 images.

For every scheme we first evaluate the quality of the derived bounding boxes in the trainval set using the CorLoc (at the standard $IoU > 0.5$ threshold and the more stricter $IoU > 0.7$) and the mIoU performance (Figure 7.1). We then train Fast R-CNN [Girshick, 2015] object detectors based on AlexNet [Krizhevsky et al., 2012] or VGG16 [Simonyan and Zisserman, 2015] from the derived boxes in the trainval set. We evaluate the object detection perfomance in the test set using the mAP metric (Figure 7.2). We also compare our schemes to a WSOL baseline (reference MIL of Sections 4.4.2, 5.4.1) and to full supervision (traditional manual way of drawing bounding boxes [Su et al., 2012]). For all our schemes we show results with crowdsourced annotations obtained on AMT.

Human verification is a very efficient scheme. At a modest extra annotation cost of only 8.9 hours, it significantly improves over the reference MIL both in terms of CorLoc and mAP. One important aspect of this scheme is that it allows us to control the quality of the generated bounding boxes in the training set. The annotators are asked to judge whether the overlap of a displayed box with an imaginary perfect box is greater than a certain threshold τ. Therefore, our scheme guarantees that the gen-

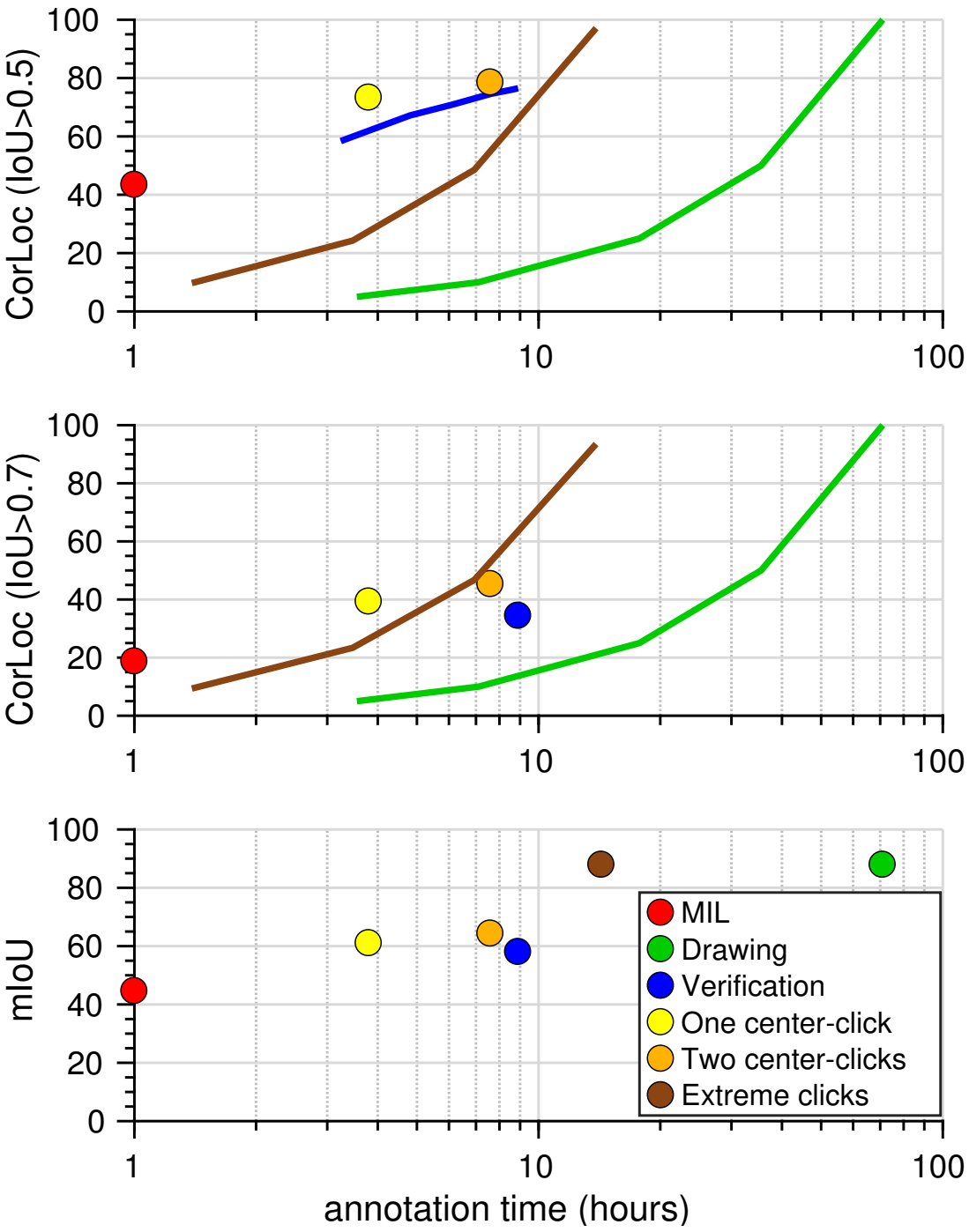


Figure 7.1: ***Evaluation of the derived bounding boxes in the PASCAL VOC trainval test.*** *We show here the the CorLoc (at the standard IoU* > 0.5 *and at the more stricter IoU* > 0.7*) and the mIoU performance against the human annotation time in hours at log scale.*

erated bounding boxes will have a quality above τ. In this thesis, we only considered $IoU > 0.5$, but stricter threshold values (e.g., $IoU > 0.7$) can be also used. This will lead to more accurate boxes, which can train better detectors at a cost of extra human

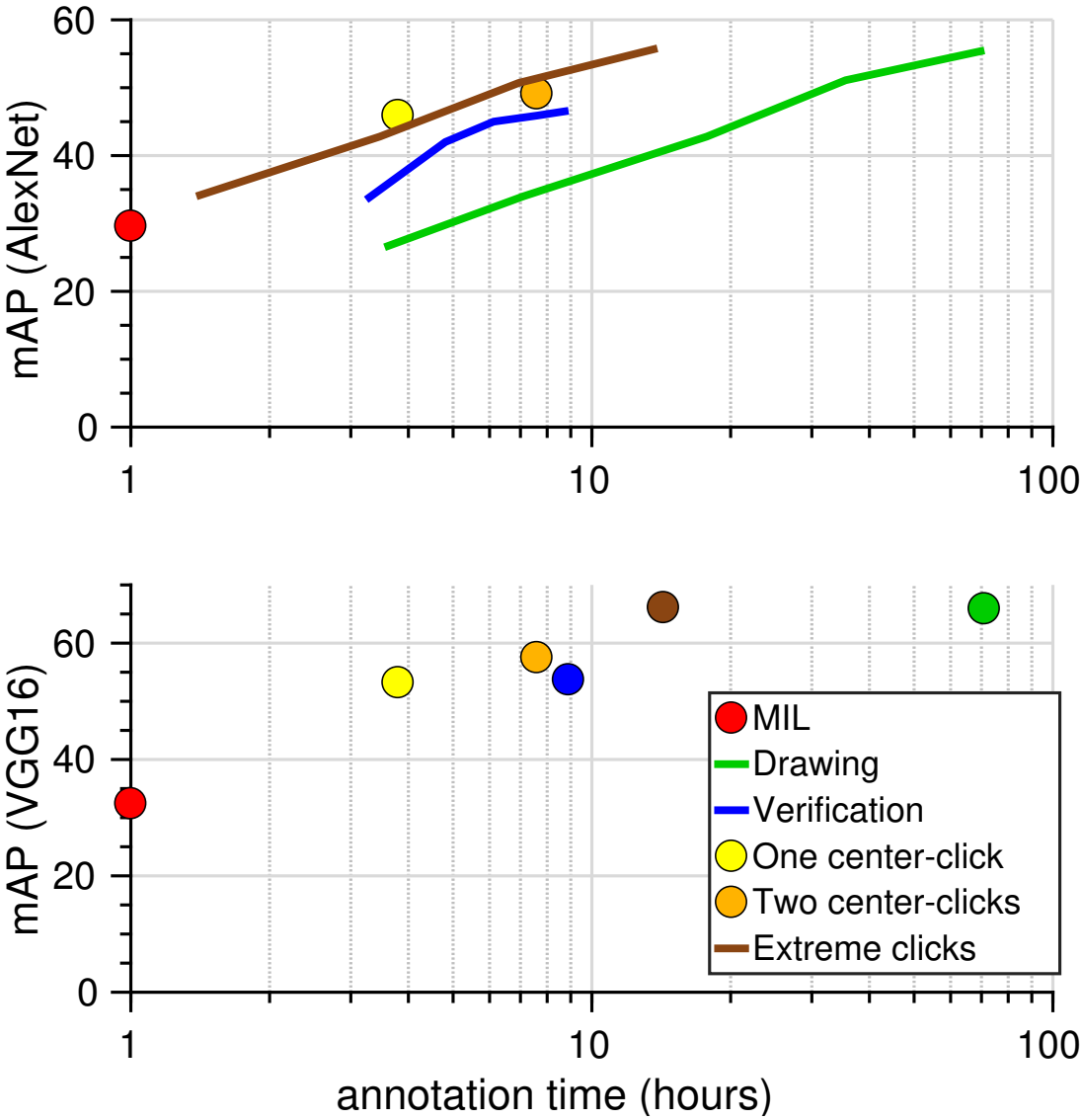


Figure 7.2: ***Evaluation of the object detection performance in the PASCAL VOC test.*** *We show here the mAP performance of object detectors trained on weak supervision, on full supervision (traditional bounding box drawing) and on our proposed schemes.*

verifications per image. Recently, Google Research generated 1.2 million accurate bounding boxes (mean IoU around 0.82) for the OpenImages dataset using our human verification scheme with $IoU > 0.7$ [Krasin et al., 2017].

Center clicking strikes a better trade-off between accuracy and efficiency than the human verification scheme. More specifically, the two-click supervision scheme leads to better CorLoc and mAP than verification. More importantly, it achieves this performance with 1.3 less hours of annotation. Given the same annotation time, one-click supervision outperforms both human verification and extreme clicking. For example,

given only 3.8 hours of annotation, one-click supervision leads to 45.9% mAP performance, 9% mAP better than verification and 3% better than extreme clicking (all using AlexNet). Therefore, in case of a small fixed annotation budget, one should consider using this scheme over the other alternatives. One limitation of the center-clicking scheme is that it does not offer a guarantee on the quality of the output bounding boxes.

Extreme clicking is an extremely accurate scheme. It is the only one scheme we propose that matches *exactly* the accuracy of the traditional bounding box drawing, both in terms of mIoU box accuracy and in terms of mAP object detection performance. More importantly, it is 5× faster. Given the same annotation time, it outperforms both human verification and two-click supervision in terms of mAP. Interestingly, training an object detector with only 50% of perfectly annotated samples leads to slightly better mAP than training with 80% good ($IoU > 0.5$) and 20% bad examples (see Figures 7.1 and 7.2 at 7 hours of annotation time).

Improve center-clicking and verification. Center-clicking and verification schemes are dependent on the MIL method on which they build on. Building on a stronger base method can significantly improve the accuracy of these schemes. Stronger WSOL methods currently exist in the literature [Bilen and Vedaldi, 2016; Kantorov et al., 2016] that lead to better CorLoc performance than our reference MIL. Also, such base methods can be made stronger by exploiting existing bounding box annotations of other classes [Guillaumin and Ferrari, 2012; Shi et al., 2012; Hoffman et al., 2014; Rochan and Wang, 2015; Uijlings et al., 2017]. Recently, the transfer learning method of Uijlings et al. [2017] has shown remarkable results. In particular, they showed that the CorLoc performance of our reference MIL can be massively increased (+27%) by such knowledge transfer. These improvements to our center-click and verification schemes could lead to a better trade-off than extreme clicking. Note that if we start from a stronger initial model, the total annotation time of the verification scheme will be decreased: more boxes will be judged as correct at the very first iteration and they will not be processed again and again at the following ones. The Google operation to generate 1.2 million accurate bounding boxes for the OpenImages dataset successfully applied this trick and showed in practice that our verification scheme enables the cheap creation of very large scale datasets with high quality boxes, reducing the need for massive annotation efforts such as ImageNet.

Complete coverage Throughout the whole thesis, we only consider localizing one

object per class per image. This helps us to keep the annotation schemes as simple as possible and enables a fair comparison with all WSOL methods. The goal of this thesis is to efficiently annotate training images that can be used to train high quality object detectors. That's why, in their current form, the proposed schemes guarantee only partial coverage of positive classes. To guarantee complete coverage, an extra step is required. One should simply ask an extra annotator to verify whether all object instances of an object class have a generated correct bounding box.

Difficulty of the dataset. Our proposed annotation schemes are dataset agnostic. Our results on COCO dataset showed that center-clicking and verification can also be applied successively in a larger and more challenging dataset with much more object classes than PASCAL VOC. Also, our human verification and extreme clicking schemes have been successfully adopted by Google Research to generate efficiently and successfully 1.2 million accurate bounding boxes for the OpenImages dataset. Therefore, it is possible that our proposed annotation schemes can be easily adapted and used to efficiently annotate other datasets that the computer vision community will be interested in the future.

A proper crowd-sourcing protocol is the key. One important aspect of our annotation schemes is the crowd-sourcing part. The quality of the annotations heavily depends on the crowd-sourcing protocol used. In all our experiments (Sections 4.5.2, 5.3 and 6.3) in Amazon Mechanical Turk (AMT), we carefully designed our crowd-sourcing in order to maximize the efficiency and the accuracy of the annotators: (1) we instructed them very carefully on how they should perform our tasks, (2) we trained them on the tasks through interactive and highly effective training stages and (3) we automatically controlled their quality while they performed our tasks (more details in Sections 4.5.2, 5.3 and 6.3). During the training stage, the annotators had to complete a qualification test, at the end of which we provided detailed feedback on how well they performed. It is also important that we made this feedback procedure fully automatic by using a few existing ground-truth annotations. This is essential as it enables high-frequency feedback iterations bypassing the need of having an expert to check the annotators results manually and resend another training task if needed. The annotators can be trained as much as needed until they can be considered as well-qualified for the annotation task. The qualification test has been proven the most effective for collecting high quality AMT annotations. Note that a badly designed crowd-sourcing protocol will lead to poor-quality annotations, which in turn will lead to low quality bounding boxes and

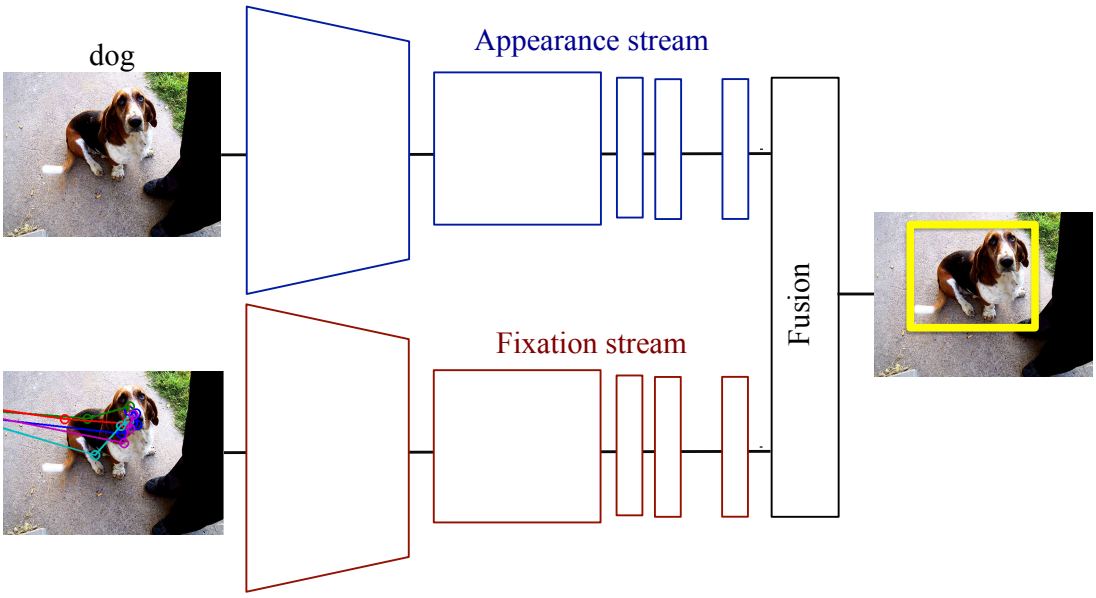


Figure 7.3: *A two-stream CNN architecture that takes as input an image, a target class and a set of eye fixations and infers the bounding box location of the object.*

low quality object detectors.

7.3 Future research and perspectives

In this section, we propose interesting directions for future research based on the methods and the experiments presented in this thesis.

Deep learning with eye tracking. In Section 3.4, we proposed a technique for deriving bounding boxes on images given eye fixations. This required us to create a set of handcrafted features (geometrical, temporal and appearance features) computed from the given set of fixations and feed them into a classifier (SVM) to learn the difference between the object and the background.

Thanks to the deep learning and the powerful CNNs, one could *learn* the features from the eye tracking data for this particular task. In particular, one could create a two-stream end-to-end trainable CNN model that takes as input an image, a target class and a set of fixations from annotators instructed to look at the image and search for an instance of the given class. The output of the network is the spatial location of the objects. One such model is illustrated in Figure 7.3. Recently, two-stream CNNs have been successfully used in different tasks such as action recognition [Simonyan and Zisserman, 2014; Gkioxari and Malik, 2015; Kalogeiton et al., 2017b,a].

During training, the network could learn more complex, non-linear relations between the object and the eye fixations than our proposed model (Section 3.4). The network should be able to learn a feature representation that is able to successfully describe the complex behavior of humans when they perform a visual search task. More importantly, it could learn to classify important patterns that occur during this task and use them to infer the whole spatial extent of the objects. For example, humans tend to fixate only on various discriminative parts of the objects (e.g. head of animals). Based on the pose and the size of the head, the network could learn where the whole body of the animal is located in the image.

Combine different annotation schemes. The annotation schemes we proposed in Chapters 4, 5 and 6 require the annotator to first search in the image, find the target object, and finally perform any of the required tasks (verification, center clicking and extreme clicking). Finding the target object takes an initial visual search time. In particular, tracking the eye movements of the annotators while they perform one of these tasks could lead to additional information about the target object (finding an object typically requires fixating it) *at no extra annotation cost*.

For example, having access to eye tracking data of the annotators while they perform the center-clicking task can give us extra points (fixations) at no extra cost. During a visual search task, some of these fixations are also on the background (see Chapter3) and it is not trivial to distinguish which of them are actually on the target object. The center click can be used to eliminate background fixations (e.g., far from center). This combination of schemes can provide us with more spatial points on the target object at no extra cost that can be again incorporated into MIL for even greater performance gains than the center clicking scheme alone.

Another interesting combination of annotations would be center clicking with verification. As shown in Section 7.2, one-click supervision is extremely fast and strikes a very good trade-off between accuracy and annotation time. Human verification steps on top of the one-click model could potentially bring a large improvement in performance, as many of the bounding boxes will be judged as correct in the first verification step. We should expect that this improvement in CorLoc and mAP will be larger than the one brought by the second center click to the model. Moreover, more complicated and interesting schemes can be introduced by this combination. For instance, iteratively alternating between center clicking and one verification could be a good alternative in case of negatively verified boxes.

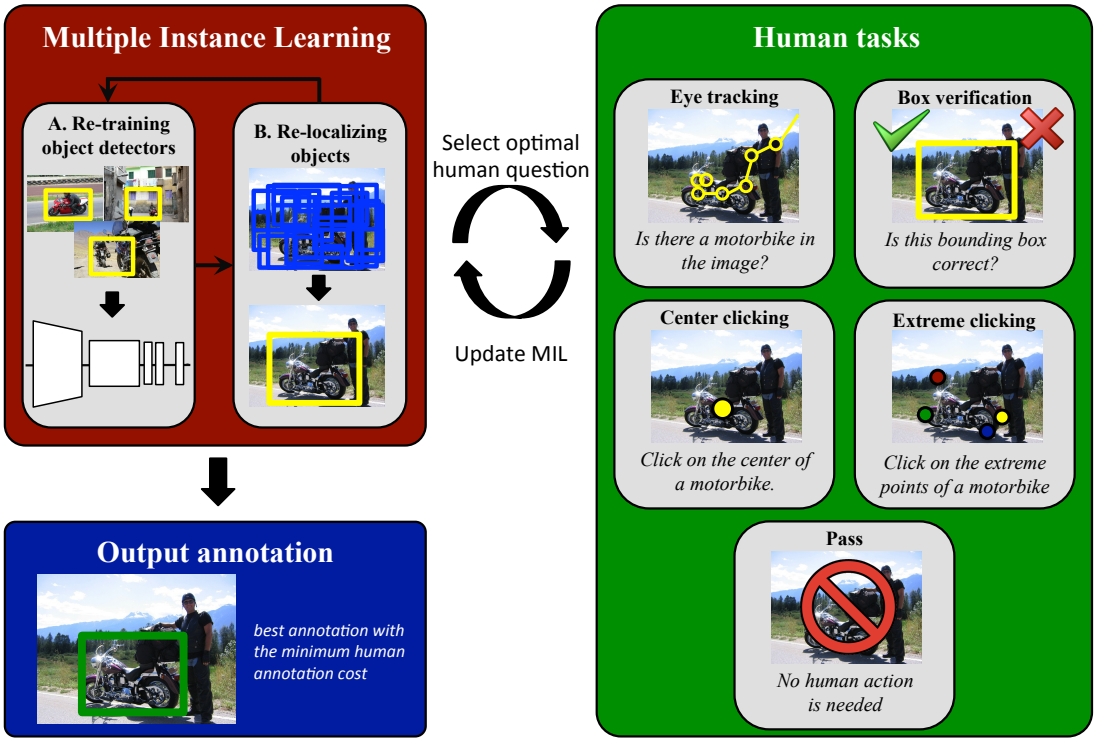


Figure 7.4: *An active learning scheme that iteratively alternates between updating the MIL framework and selecting the optimal question to query the annotators.*

Active learning. The goal of all our proposed annotation schemes was to obtain high quality object detectors by cutting the unit cost of producing one bounding box on an object. This results in savings in annotation time. However, annotating very large-scale datasets (in the million range) with one of our schemes would still require millions of cheap annotations, which would still result in lots of annotation hours. Active learning could be used to further reduce the overall annotation time.

Active learning schemes iteratively train models while requesting humans to annotate only a subset of the data points actively selected by the learner as being the most informative. Previous active learning work has mainly focused on image classification [Joshi et al., 2009; Kapoor et al., 2007; Kovashka et al., 2011; Qi et al., 2008], and free-form region labeling [Siddiquie and Gupta, 2010; Vijayanarasimhan and Grauman, 2008, 2009]. In the task of object detection, a few researchers have proposed active learning strategies that can produce high quality detectors, but they still require humans to draw lots of bounding-boxes in order to get there (about one third of the boxes), leading to limited gains in terms of total annotation time [Vijayanarasimhan and Grauman, 2014; Yao et al., 2012].

Such techniques could be used in conjunction with our schemes to further reduce

the annotation cost. Objects that are already correctly localized by the initial weakly supervised localization algorithm do not need to be ever annotated using any of our schemes. Also, objects that are still wrongly localized even after using our annotations should also not be annotated. For example, in our verification scheme, there are some very difficult objects that require more than 15 verifications before reaching a good localization. This procedure would require an extra component to automatically decide which examples should be sent to humans for annotation.

Another extension of the above scheme that also combines all our proposed schemes is illustrated in Figure 7.4. The input to this system is a set of images to be annotated with bounding boxes and a desired budget. The output is a set of accurate object bounding boxes. The goal of this system is to minimize the total human annotation cost needed to obtain accurate bounding boxes. The system alternates between updating the appearance model (e.g., one full iteration of WSOL MIL) and asking humans to provide a type of annotation for an image. In some very easy cases, the output of MIL is already very accurate and no further annotation is needed. There are also cases where a very cheap annotation (one center click, one verification) could lead to a very accurate bounding box or there are cases where that would fail (center clicking does not lead to a good box or we need to verify 20 different wrong boxes before we find a good one). In those cases, one should better use extreme clicking.

Video annotation. Drawing bounding boxes in videos is more expensive than in still images as this requires an annotation in every video frame. Note that annotating a dog running in a small video of one minute length would require more than one thousand bounding boxes. Large video datasets (such as YTO [Prest et al., 2012], UCF-101 [Soomro et al., 2012] or YouTube-8M [Abu-El-Haija et al., 2016]) contain millions or even billions of video frames. At these scales, even our efficient center-clicking or verification schemes could not solve this annotation problem alone.

Given the temporal continuity of videos, one could propagate the bounding boxes of a moving object to the next or previous frames by leveraging the object motion in neighbouring frames. For this reason, several tracking algorithms have been proposed in the literature [Andriluka et al., 2008; Babenko et al., 2009; Benfold and Reid, 2011; Berclaz et al., 2011; Grabner et al., 2008; Henschel et al., 2017; Leibe et al., 2007; Sadeghian et al., 2017; Tang et al., 2017]. These approaches, however, do not perform well in challenging videos, e.g. with cluttered scenes, with partially or fully occluded objects, or with significant appearance variations.

Vondrick et al. [2013] designed an interactive video annotation interface, where the annotators can play the video, draw bounding boxes around the objects while the algorithm tracks each object through the video. If the tracking algorithm fails, the annotator can still intervene and fix the object bounding box. This scheme could lead to high quality annotations for the whole video without drawing drawing boxes in every single frame. It still requires, however, the annotator to scan the whole video slowly and intervene several times.

Therefore, an interesting direction would be designing a system that automatically selects *which* video frames to annotate. Such a system could potentially lead to great saving in annotation time, as videos are redundant. Moreover, these selected frames could be annotated with one of our proposed schemes bypassing the need for drawing *any box* in the video. This system can be seen as an extension of the active learning scheme described above (Figure 7.4). Taking into account the temporal relation of the frames, the system should be able to predict whether an annotation in a specific frame can be correctly propagated to the following ones.

Finally, it would be an interesting direction to collect eye-tracking data on video, thus extending our work of Chapter 3. Ideally, some day this could lead to the dream paradigm of "training object detectors while watching TV".

Appendix A

Qualitative examples of extreme clicking

Figure A.1: **Qualitative examples of extreme clicking annotations on PASCAL VOC 2007 trainval set.** For each example, we provide the extreme clicking bounding-box (blue box) and the exact positions of the extreme clicks (green dots), the PASCAL ground-truth bounding-boxes (yellow box) and the exact IoU between the two annotations.

Figure A.1 shows qualitative results on various PASCAL VOC 2007 trainval images comparing extreme clicking with the original ground-truth (GT Boxes). As shown

in Table 6.2, 94% of the extreme clicking boxes have high IoU (IoU>0.7) with the corresponding GT Box. Only for 6% of all objects the two annotations are considerably different (IoU<0.7). In order to fully understand why in these few cases the extreme clicks and PASCAL annotations diverge, we manually inspected annotations for 50 randomly picked samples with almost no overlap (IoU<0.3) and 100 samples with low overlap (IoU $\in$ [0.3,0.7]).

We found out that in cases with almost no overlap, 62% of our annotations are correct but those objects are not annotated in the PASCAL ground-truth; 18% annotations are on objects of a similar class (e.g. a side-table instead of a dining-table, a vase with flowers instead of a potted-plant, a pickup truck instead of a car); 14% are on an entirely wrong class; 6% are spatial annotation errors (e.g. missing a part of the object).

For low overlap cases there are errors either in extreme click annotations or in the PASCAL annotations. The majority of these cases are partially occluded objects, where small parts of an object (e.g. hand, foot, tail, leg, wheel) appear quite far from the main visible part. Other cases are objects with thin parts like antennas and masts, yet others are dark images. In 21% of these low overlap cases, extreme clicks provided better annotations, in 23% the PASCAL ground-truth was better, and in 56% of the cases we could not decide which annotation was better (these were mostly small objects).

Appendix B

Acronyms

AMT - Amazon Mechanical Turk

AP - Average Precision

BoW - Bag of Words

CNN - Convolutional Neural Network

CorLoc - Correct Localization

CRF - Conditional Random Field

DPM - Deformable Part Model

FS - Full supervision

HoG - Histogram of Oriented Gradients

GMM - Gaussian Mixture Model

GT - Ground-truth

ILSVRC - Imagenet Large Scale Visual Recognition Challenge

IoU - Intersection-over-Union

mAP - mean Average Precision

MIL - Multiple Instance Learning

mIoU - mean Intersection-over-Union

SGD - Stochastic Gradient Descent

SIFT - Scale-Invariant Feature Transform

SPM - Spatial Pyramid Matching

SSD - Single Shot Detector

SVM - Support Vector Machine

R-CNN - Region-based Convolutional Neural Network

ReLU - Rectified Linear Unit

ResNet - Residual Network

RoI - Region of Interest

RPN - Region Proposal Network

WS - Weak Supervision

WSOL - Weakly Supervised Object Localization

YPCMM - Yes/Part/Container/Mixed/Missed

Bibliography

Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, P., Toderici, G., Varadarajan, B., and Vijayanarasimhan, S. (2016). Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*.

Alexe, B., Deselaers, T., and Ferrari, V. (2010). What is an object? In *CVPR*.

Alexe, B., Deselaers, T., and Ferrari, V. (2012). Measuring the objectness of image windows. *IEEE Trans. on PAMI*.

Andriluka, M., Pishchulin, L., Gehler, P., and Schiele, B. (2014). 2d human pose estimation: New benchmark and state of the art analysis. In *CVPR*.

Andriluka, M., Roth, S., and Schiele, B. (2008). People-tracking-by-detection and people-detection-by-tracking. In *CVPR*.

Arbeláez, P., Maire, M., Fowlkes, C., and Malik, J. (2011). Contour detection and hierarchical image segmentation. *IEEE Trans. on PAMI*.

Arbeláez, P., Pont-Tuset, J., Barron, J. T., Marques, F., and Malik, J. (2014). Multiscale combinatorial grouping. In *CVPR*.

Aytar, Y. and Zisserman, A. (2011). Tabula rasa: Model transfer for object category detection. In *ICCV*.

Aytar, Y. and Zisserman, A. (2012). Enhancing exemplar SVMs using part level transfer regularization. In *BMVC*.

Babenko, B., Yang, M.-H., and Belongie, S. (2009). Visual tracking with online multiple instance learning. In *CVPR*.

Bai, X. and Sapiro, G. (2009). Geodesic matting: A framework for fast interactive image and video segmentation and matting. *IJCV*.

Bawa, P. (2017). Object localization using a portable eye-tracker. BSc dissertation, University of Edinburgh.

Bay, H., Ess, A., Tuytelaars, T., and van Gool, L. (2008). SURF: Speeded up robust features. *CVIU*.

Bearman, A., Russakovsky, O., Ferrari, V., and Fei-Fei, L. (2016). What's the point: Semantic segmentation with point supervision. In *ECCV*.

Bell, S., Upchurch, P., Snavely, N., and Bala, K. (2015). Material recognition in the wild with the materials in context database. In *CVPR*.

Benfold, B. and Reid, I. (2011). Stable multi-target tracking in real-time surveillance video. In *CVPR*.

Berclaz, J., Fleuret, F., Turetken, E., and Fua, P. (2011). Multiple object tracking using k-shortest paths optimization. *IEEE Trans. on PAMI*, 33(9).

Berg, T., Berg, A., Edwards, J., Mair, M., White, R., Teh, Y., Learned-Miller, E., and Forsyth, D. (2004). Names and Faces in the News. In *CVPR*.

Bilen, H., Pedersoli, M., and Tuytelaars, T. (2014). Weakly supervised object detection with posterior regularization. In *BMVC*.

Bilen, H., Pedersoli, M., and Tuytelaars, T. (2015). Weakly supervised object detection with convex clustering. In *CVPR*.

Bilen, H. and Vedaldi, A. (2016). Weakly supervised deep detection networks. In *CVPR*.

Biswas, A. and Parikh, D. (2013). Simultaneous active learning of classifiers & attributes via relative feedback. In *CVPR*.

Blake, A., Rother, C., Brown, M., Perez, P., and Torr, P. (2004). Interactive image segmentation using an adaptive GMMRF model. In *ECCV*.

Blaschko, M. B., Vedaldi, A., and Zisserman, A. (2010). Simultaneous object detection and ranking with weak supervision. In *NIPS*.

Blot, M., Cord, M., and Thome, N. (2016). Max-min convolutional neural networks for image classification. In *ICIP*. IEEE.

Bojanowski, P., Bach, F., Laptev, I., Ponce, J., Schmid, C., and Sivic, J. (2013). Finding actors and actions in movies. In *ICCV*.

Boykov, Y. and Jolly, M. P. (2001). Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *ICCV*.

Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on PAMI*, 26(9):1124–1137.

Brainard, D. H. (1997). The Psychophysics Toolbox. *Spatial Vision*, 10:433–436.

Branson, S., Perona, P., and Belongie, S. (2011). Strong supervision from weak annotation: Interactive training of deformable part models. In *ICCV*.

Branson, S., Wah, C., Schroff, F., Babenko, B., Welinder, P., Perona, P., and Belongie, S. (2010). Visual recognition with humans in the loop. In *ECCV*.

Carreira, J. and Sminchisescu, C. (2010). Constrained parametric min-cuts for automatic object segmentation. In *CVPR*.

Carreira, J. and Sminchisescu, C. (2012). CPMC: Automatic object segmentation using constrained parametric min-cuts. *IEEE Trans. on PAMI*.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. (2015). Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*.

Cheng, M.-M., Zhang, Z., Lin, W.-Y., and Torr, P. (2014). Bing: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*.

Chum, O. and Zisserman, A. (2007). An exemplar model for learning object classes. In *CVPR*.

Cinbis, R., Verbeek, J., and Schmid, C. (2014). Multi-fold mil training for weakly supervised object localization. In *CVPR*.

Cinbis, R., Verbeek, J., and Schmid, C. (2016). Weakly supervised object localization with multi-fold multiple instance learning. *IEEE Trans. on PAMI*.

Crandall, D. J. and Huttenlocher, D. (2006). Weakly supervised learning of part-based spatial models for visual object recognition. In *ECCV*.

CrowdFlower (2016). Crowdflower bounding box annotation tool. https://www.youtube.com/watch?v=lUIU2_HW4Ic.

Csurka, G., Bray, C., Dance, C., and Fan, L. (2004a). Visual categorization with bags of keypoints. In *ECCV Workshop on Statistical Learning in Computer Vision*.

Csurka, G., Bray, C., Dance, C., and Fan, L. (2004b). Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22.

Dalal, N. and Triggs, B. (2005). Histogram of Oriented Gradients for human detection. In *CVPR*.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *CVPR*.

Deng, J., Krause, J., and Fei-Fei, L. (2013). Fine-grained crowdsourcing for fine-grained recognition. In *CVPR*.

Deselaers, T., Alexe, B., and Ferrari, V. (2010). Localizing objects while learning their appearance. In *ECCV*.

Deselaers, T., Alexe, B., and Ferrari, V. (2012). Weakly supervised localization and learning with generic knowledge. *IJCV*.

Dietterich, T. G., Lathrop, R. H., and Lozano-Perez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71.

Dollar, P. and Zitnick, C. (2013). Structured forests for fast edge detection. In *ICCV*.

Dollar, P. and Zitnick, C. (2014). Edge boxes: Locating object proposals from edges. In *ECCV*.

Duchenne, O., Audibert, J.-Y., Keriven, R., Ponce, J., and Ségonne, F. (2008). Segmentation by transduction. In *CVPR*.

Durand, T., Mordan, T., Thome, N., and Cord, M. (2017). Wildcat: Weakly supervised learning of deep convnets for image classification, pointwise localization and segmentation. In *CVPR*.

Duygulu, P., Barnard, K., de Freitas, N., and Forsyth, D. (2002). Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*.

Ehinger, K. A., Hidalgo-Sotelo, B., Torralba, A., and Olivia, A. (2009). Modelling search for people in 900 scenes: A combined source model of eye guidance. *Visual Cognition*.

Einhäuser, W., Spain, M., and Perona, P. (2008). Objects predict fixations better than early saliency. *Journal of Vision*, 8:1–26.

Endres, I., Farhadi, A., Hoiem, D., and Forsyth, D. A. (2010). The benefits and challenges of collecting richer object annotations. In *DeepVision workshop at CVPR*.

Endres, I. and Hoiem, D. (2010). Category independent object proposals. In *ECCV*.

Everingham, M., Sivic, J., and Zisserman, A. (2006). Hello! my name is ... buffy - automatic naming of characters in tv video. In *BMVC*.

Everingham, M., Van Gool, L., Williams, C., Winn, J., and Zisserman, A. (2007). The PASCAL Visual Object Classes Challenge 2007 Results.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2010). The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. (2012). The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Fei-Fei, L., Fergus, R., and Perona, P. (2007). Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *cviu*.

Felzenszwalb, P., Girshick, R., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part based models. *IEEE Trans. on PAMI*, 32(9).

Felzenszwalb, P. and Huttenlocher, D. (2005). Pictorial structures for object recognition. *IJCV*, 61(1).

Fergus, R., Perona, P., and Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *CVPR*.

Ferrari, V., Jurie, F., and Schmid, C. (2010). From images to shape models for object detection. *IJCV*, 87(3).

Ferrari, V., Marin, M., and Zisserman, A. (2008). Progressive search space reduction for human pose estimation. In *CVPR*.

Ferrari, V., Tuytelaars, T., and Van Gool, L. (2004). Simultaneous object recognition and segmentation by image exploration. In *ECCV*.

Fischler, M. and Elschlager, R. (1973). The representation and matching of pictorial structures. *IEEE Transactions on Computer*, c-22(1):67–92.

Freedman, D. and Zhang, T. (2005). Interactive graph cut based segmentation with shape priors. In *CVPR*.

Galleguillos, C., Babenko, B., Rabinovich, A., and Belongie, S. (2008). Weakly supervised object localization with stable segmentations. In *ECCV*.

Gidaris, S. and Komodakis, N. (2015). Object detection via a multi-region and semantic segmentation-aware cnn model. In *ICCV*.

Gidaris, S. and Komodakis, N. (2016). Locnet: Improving localization accuracy for object detection. In *CVPR*.

Girshick, R. (2015). Fast R-CNN. In *ICCV*.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*.

Gkioxari, G. and Malik, J. (2015). Finding action tubes. In *CVPR*.

Grabner, H., Leistner, C., and Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. *ECCV*.

Grady, L. (2006). Random walks for image segmentation. *IEEE Trans. on PAMI*, 28(11):1768–1783.

Guillaumin, M. and Ferrari, V. (2012). Large-scale knowledge transfer for object localization in imagenet. In *CVPR*.

Guillaumin, M., Küttel, D., and Ferrari, V. (2014). ImageNet auto-annotation with segmentation propagation. *IJCV*.

Gulshan, V., Rother, C., Criminisi, A., Blake, A., and Zisserman, A. (2010). Geodesic star convexity for interactive image segmentation. In *CVPR*.

Gupta, A. and Davis, L. (2008). Beyond nouns: Exploiting prepositions and comparators for learning visual classifiers. In *ECCV*.

Harel, J., Koch, C., and Perona, P. (2007). Graph-based visual saliency. In *NIPS*.

Hariharan, B., Arbeláez, P., Bourdev, L., Maji, S., and Malik, J. (2011). Semantic contours from inverse detectors. In *ICCV*.

Harzallah, H., Jurie, F., and Schmid, C. (2009). Combining efficient object localization and image classification. In *ICCV*.

Hata, K., Krishna, R., Fei-Fei, L., and Bernstain, M. (2017). A glimpse far into the future: Understanding long-term crowd worker accuracy. In *CSCW*.

He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*.

Henderson, J. (2003). Human gaze control in real-world scene perception. *Trends in Cognitive Sciences*, 7:498–504.

Henschel, R., Leal-Taixé, L., Cremers, D., and Rosenhahn, B. (2017). Improvements to frank-wolfe optimization for multi-detector multi-object tracking. *arXiv preprint arXiv:1705.08314*.

Hoffman, J., Guadarrama, S., Tzeng, E., Hu, R., and Donahue, J. (2014). LSDA: Large scale detection through adaptation. In *NIPS*.

Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*.

Itti, L., Koch, C., and Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Trans. on PAMI*, 20(11):1254–1259.

Jain, S. and Grauman, K. (2016a). Click carving: Segmenting objects in video with point clicks. In *Proceedings of the Fourth AAAI Conference on Human Computation and Crowdsourcing*.

Jain, S. D. and Grauman, K. (2013). Predicting sufficient annotation strength for interactive foreground segmentation. In *ICCV*.

Jain, S. D. and Grauman, K. (2016b). Active image segmentation propagation. In *CVPR*.

Jerripothula, K. R., Cai, J., and Yuan, J. (2016). Cats: co-saliency activated tracklet selection for video co-localization. In *ECCV*.

Jia, Y. (2013). Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org/.

Jiang, M., Huang, S., Duan, J., and Zhao, Q. (2015). Salicon: Saliency in context. In *CVPR*.

Johnson, S. and Everingham, M. (2010). Clustered pose and nonlinear appearance models for human pose estimation. In *BMVC*.

Johnson, S. and Everingham, M. (2011). Learning effective human pose estimation from inaccurate annotation. In *CVPR*.

Joshi, A. J., Porikli, F., and Papanikolopoulos, N. (2009). Multi-class active learning for image classification. In *CVPR*.

Judd, T., Ehinger, K., Durand, F., and Torralba, A. (2009). Learning to predict where humans look. In *IEEE International Conference on Computer Vision (ICCV)*.

Kalogeiton, V., Ferrari, V., and Schmid, C. (2016). Analysing domain shift factors between videos and images for object detection. *IEEE Trans. on PAMI*.

Kalogeiton, V., Weinzaepfel, P., Ferrari, V., and Schmid, C. (2017a). Action tubelet detector for spatio-temporal action localization. *ICCV*.

Kalogeiton, V., Weinzaepfel, P., Ferrari, V., and Schmid, C. (2017b). Joint learning of object and action detectors. In *ICCV*.

Kantorov, V., Oquab, M., Cho, M., and Laptev, I. (2016). Contextlocnet: Context-aware deep network models for weakly supervised localization. In *ECCV*.

Kapoor, A., Grauman, K., Urtasun, R., and Darrell, T. (2007). Active learning with gaussian processes for object categorization. In *ICCV*.

Karthikeyan, S., Jagadeesh, V., Shenoy, R., Eckstein, M., and Manjunath, B. (2013). From where and how to what we see. In *ICCV*.

Kim, G. and Torralba, A. (2009). Unsupervised detection of regions of interest using iterative link analysis. In *NIPS*.

Kosslyn, S. M., Thompson, W. L., Kim, I. J., and Alpert, N. M. (1995). Topographic representations of mental images in primary visual cortex. *Nature*, 378(6556):496–498.

Kovashka, A. and Grauman, K. (2015). Discovering attribute shades of meaning with the crowd. *IJCV*.

Kovashka, A., Vijayanarasimhan, S., and Grauman, K. (2011). Actively selecting annotations among objects and attributes. In *ICCV*.

Krafka, K., Khosla, A., Kellnhofer, P., Kannan, H., Bhandarkar, S., Matusik, W., and Torralba, A. (2016). Eye tracking for everyone. In *CVPR*.

Krähenbühl, P. and Koltun, V. (2014). Geodesic object proposals. In *ECCV*.

Krasin, I., Duerig, T., Alldrin, N., Ferrari, V., Abu-El-Haija, S., Kuznetsova, A., Rom, H., Uijlings, J., Popov, S., Veit, A., Belongie, S., Gomes, V., Gupta, A., Sun, C., Chechik, G., Cai, D., Feng, Z., Narayanan, D., and Murphy, K. (2017). Openimages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. (2013). 3d object representations for fine-grained categorization. In *ICCV Workshop on 3D Representation and Recognition*.

Krishna, R. A., Hata, K., Chen, S., Kravitz, J., Shamma, D. A., Fei-Fei, L., and Bernstein, M. S. (2016). Embracing error to enable rapid crowdsourcing. In *CHI*.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *NIPS*.

Kruthiventi, S. S., Ayush, K., and Babu, R. V. (2017). Deepfix: A fully convolutional neural network for predicting human eye fixations. *IEEE Transactions on Image Processing*.

Kuettel, D. and Ferrari, V. (2012). Figure-ground segmentation by transferring window masks. In *CVPR*.

Kuettel, D., Guillaumin, M., and Ferrari, V. (2012). Segmentation Propagation in ImageNet. In *ECCV*.

Kumar Singh, K., Xiao, F., and Jae Lee, Y. (2016). Track and transfer: Watching videos to simulate strong human supervision for weakly-supervised object detection. In *CVPR*.

Lad, S. and Parikh, D. (2014). Interactively guiding semi-supervised clustering via attribute-based explanations. In *ECCV*.

Ladicky, L., Russell, C., and Kohli, P. (2009). Associative hierarchical CRFs for object class image segmentation. In *ICCV*.

Lampert, C., Nickisch, H., and Harmeling, S. (2009). Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*.

Laptev, I., Marszałek, M., Schmid, C., and Rozenfeld, B. (2008). Learning realistic human actions from movies. In *CVPR*.

Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *CVPR*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, Y. and Grauman, K. (2009). Shape discovery from unlabeled image collections. In *CVPR*.

Leibe, B., Schindler, K., and Van Gool, L. (2007). Coupled detection and trajectory estimation for multi-object tracking. In *ICCV*.

Leistner, C., Godec, M., Schulter, S., Saffari, A., and Bischof, H. (2011). Improving classifiers with weakly-related videos. In *CVPR*.

Lempitsky, V., Kohli, P., Rother, C., and Sharp, T. (2009). Image segmentation with a bounding box prior. In *ICCV*.

Levinshtein, A., Stere, A., Kutulakos, K., Fleed, D., and Dickinson, S. (2009). Turbopixels: Fast superpixels using geometric flows. In *IEEE Trans. on PAMI*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. (2014). Microsoft COCO: Common objects in context. In *ECCV*.

Liu, N., Han, J., Zhang, D., Wen, S., and Liu, T. (2015). Predicting eye fixations using convolutional neural networks. In *CVPR*.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *ECCV*.

Lowe, D. (1999). Object recognition from local scale-invariant features. In *ICCV*, volume 2, pages 1150–1157.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110.

Luo, J., Caputo, B., and Ferrari, V. (2009). Who's doing what: Joint modeling of names and verbs for simultaneous face and pose annotation. In *NIPS*.

Mahamud, S. and Hebert, M. (2003). The optimal distance measure for object detection. In *CVPR*, volume 1, pages 248–258.

Maji, S., Berg, A. C., and Malik, J. (2008). Classification using intersection kernel support vector machines is efficient. In *CVPR*.

Malisiewicz, T., Gupta, A., and Efros, A. (2011). Ensemble of exemplar-svms for object detection and beyond. In *ICCV*.

Manen, S., Guillaumin, M., and Van Gool, L. (2013). Prime object proposals with randomized prim's algorithm. In *ICCV*.

Marszalek, M., Laptev, I., and Schmid, C. (2009). Actions in context. In *cvpr*.

Matas, J., Chum, O., Urban, M., and Pajdla, T. (2002). Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, pages 384–393.

Mathe, S. and Sminchisescu, C. (2012). Dynamic eye movement datasets and learnt saliency models for visual action recognition. In *ECCV*.

Mettes, P., van Gemert, J. C., and Snoek, C. G. (2016). Spot on: Action localization from pointly-supervised proposals. In *ECCV*.

Mikolajczyk, K. and Schmid, C. (2004). Scale & affine invariant interest point detectors. *IJCV*, 1(60):63–86.

Mishra, A., Aloimonos, Y., and Fah, C. L. (2009). Active segmentation with fixation. In *ICCV*.

Monsell, S. (2003). Task switching. *Trends in Cognitive Sciences*, 7(3):134–140.

Moreels, P., Maire, M., and Perona, P. (2004). Recognition by probabilistic hypothesis construction. In *ECCV*, pages 55–68.

Murase, H. and Nayar, S. (1995). Visual learning and recognition of 3D objects from appearance. *IJCV*, 14(1).

Nguyen, M., Torresani, L., de la Torre, F., and Rother, C. (2009). Weakly supervised discriminative localization and classification: a joint learning process. In *ICCV*.

Noh, H., Hong, S., and Han, B. (2015). Learning deconvolution network for semantic segmentation. In *ICCV*.

Nuthmann, A. and Henderson, J. M. (2010). Object-based attentional selection in scene viewing. *Journal of Vision*, 10(8):1–19.

Oneata, D., Revaud, J., Verbeek, J., and Schmid, C. (2014). Spatio-temporal object detection proposals. In *ECCV*.

Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2015). Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *CVPR*.

Pandey, M. and Lazebnik, S. (2011). Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV*.

Papadopoulos, D. P., Clarke, A. D. F., Keller, F., and Ferrari, V. (2014). Training object class detectors from eye tracking data. In *ECCV*.

Papadopoulos, D. P., Uijlings, J. R., Keller, F., and Ferrari, V. (2017). Training object class detectors with click supervision. In *CVPR*.

Papadopoulos, D. P., Uijlings, J. R. R., Keller, F., and Ferrari, V. (2016). We don't need no bounding-boxes: Training object class detectors using only human verification. In *CVPR*.

Papandreou, G., Chen, L.-C., Murphy, K., and Yuille, A. L. (2015). Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV*.

Papoutsaki, A., Sangkloy, P., Laskey, J., Daskalova, N., Huang, J., and Hays, J. (2016). Webgazer: Scalable webcam eye tracking using user interactions. In *IJCAI*.

Parikh, D. and Grauman, K. (2011). Relative attributes. In *ICCV*.

Parkash, A. and Parikh, D. (2012). Attributes for classifier feedback. In *ECCV*.

Platt, J. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*.

Prest, A., Leistner, C., Civera, J., Schmid, C., and Ferrari, V. (2012). Learning object class detectors from weakly annotated video. In *CVPR*.

Price, B. L., Morse, B., and Cohen, S. (2010). Geodesic graph cut for interactive image segmentation. In *CVPR*.

Qi, G.-J., Hua, X.-S., Rui, Y., Tang, J., and Zhang, H.-J. (2008). Two-dimensional active learning for image classification. In *CVPR*.

Rahtu, E., Kannala, J., and Blaschko, M. (2011). Learning a Category Independent Object Detection Cascade. In *ICCV*.

Ramanan, D. (2006). Learning to parse images of articulated bodies. In *NIPS*.

Ramanathan, S., Katti, H., Sebe, N., Kankanhalli, M., and Chua, T.-S. (2010). An eye fixation database for saliency detection in images. In *ECCV*.

Rantalankila, P., Kannala, J., and Rahtu, E. (2014). Generating object segmentation proposals using global and local search. In *CVPR*.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *CVPR*.

Redmon, J. and Farhadi, A. (2016). Yolo9000: better, faster, stronger. In *CVPR*.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*.

Ristin, M., Gall, J., and Van Gool, L. (2012). Local context priors for object proposal generation. In *ACCV*.

Rochan, M. and Wang, Y. (2015). Weakly supervised localization of novel objects using appearance transfer. In *CVPR*.

Rother, C., Kolmogorov, V., and Blake, A. (2004). Grabcut: interactive foreground extraction using iterated graph cuts. *SIGGRAPH*.

Rothganger, F., Lazebnik, S., Schmid, C., and Ponce, J. (2003). 3D object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *CVPR*.

Rubinstein, J. S., Meyer, D. E., and Evans, J. E. (2001). Executive control of cognitive processes in task switching. *Journal of Experimental Psycholgy: Human Perception and Performance*, 27(4):763–797.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., and Fei-Fei, L. (2015a). ImageNet large scale visual recognition challenge. *IJCV*.

Russakovsky, O., Li, L.-J., and Fei-Fei, L. (2015b). Best of both worlds: human-machine collaboration for object annotation. In *CVPR*.

Russakovsky, O., Lin, Y., Yu, K., and Fei-Fei, L. (2012). Object-centric spatial pooling for image classification. In *ECCV*.

Russell, B. C., Murphy, K. P., and Freeman, W. T. (2008). LabelMe: a database and web-based tool for image annotation. *IJCV*.

Sadeghian, A., Alahi, A., and Savarese, S. (2017). Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *arXiv preprint arXiv:1701.01909*.

Salakhutdinov, R., Torralba, A., and Tenenbaum, J. (2011). Learning to share visual appearance for multiclass object detection. In *CVPR*.

Sapp, B. and Taskar, B. (2013). Modec: Multimodal decomposable models for human pose estimation. In *CVPR*.

Schmid, C. and Mohr, R. (1996). Combining greyvalue invariants with local constraints for object recognition. Technical report, INRIA Rhône-Alpes, Grenoble, France.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., and LeCun, Y. (2014). Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*.

Shapovalova, N., Vahdat, A., Cannons, K., Lan, T., and Mori, G. (2012). Similarity constrained latent support vector machine: An application to weakly supervised action classification. In *ECCV*.

Shepard, R. N. and Metzler, J. (1971). Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703.

Shi, Z., Hospedales, T., and Xiang, T. (2015). Bayesian joint modelling for object localisation in weakly labelled images. *IEEE Trans. on PAMI*.

Shi, Z., Hospedales, T. M., and Xiang, T. (2013). Bayesian joint topic modelling for weakly supervised object localisation. In *ICCV*.

Shi, Z., Siva, P., and Xiang, T. (2012). Transfer learning by ranking for weakly supervised object annotation. In *BMVC*.

Siddiquie, B. and Gupta, A. (2010). Beyond active noun tagging: Modeling contextual interactions for multi-class active learning. In *CVPR*.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Deep inside convolutional networks: Visualising image classification models and saliency maps. In *ICLR workshop*.

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *NIPS*, pages 568–576.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In *ICLR*.

Siva, P., Russell, C., Xiang, T., and Agapito, L. (2013). Looking beyond the image: Unsupervised learning for object saliency and detection. In *CVPR*.

Siva, P. and Xiang, T. (2011). Weakly supervised object detector learning with model drift detection. In *ICCV*.

Siva, P., Xiang, T., and Russell, C. (2012). In defence of negative mining for annotating weakly labeled data. In *ECCV*.

Sivic, J. and Zisserman, A. (2003a). Video Google: A text retrieval approach to object matching in videos. In *ICCV*.

Sivic, J. and Zisserman, A. (2003b). Video Google: A text retrieval approach to object matching in videos. In *ICCV*, volume 2, pages 1470–1477.

Skovsgaard, H., Hansen, J. P., and Møllenbach, E. (2013). Gaze tracking through smartphones. In *Gaze Interaction in the Post-WIMP World CHI 2013 One-day Workshop*.

Song, H., Girshick, R., Jegelka, S., Mairal, J., Harchaoui, Z., and Darell, T. (2014a). On learning to localize objects with minimal supervision. In *ICML*.

Song, H., Lee, Y., Jegelka, S., and Darell, T. (2014b). Weakly-supervised discovery of visual pattern configurations. In *NIPS*.

Soomro, K., Zamir, A. R., and Shah, M. (2012). UCF101: A dataset of 101 human action classes from videos in the wild. Technical Report CRCV-TR-12-01, University of Central Florida.

Sorokin, A. and Forsyth, D. (2008). Utility data annotation with amazon mechanical turk. In *Workshop at CVPR*.

Soukoreff, R. W. and MacKenzie, I. S. (2004). Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. *International Journal of Human-Computer Studies*, 61(6):751–789.

Spare5/MightyAI (2017). Bounding box drawing instruction video. https://www.youtube.com/watch?v=3SZyFJiMGOw.

Su, H., Deng, J., and Fei-Fei, L. (2012). Crowdsourcing annotations for visual object detection. In *AAAI Human Computation Workshop*.

Sun, C., Paluri, M., Collobert, R., Nevatia, R., and Bourdev, L. (2016). Pronet: Learning to propose object-specific boxes for cascaded neural networks. In *CVPR*.

Sun, Q. and Batra, D. (2015). Submodboxes: Near-optimal search for a set of diverse object proposals. In *NIPS*, pages 1378–1386.

Tang, K., Joulin, A., Li, L.-J., and Fei-Fei, L. (2014). Co-localization in real-world images. In *CVPR*.

Tang, K., Sukthankar, R., Yagnik, J., and Fei-Fei, L. (2013). Discriminative segment annotation in weakly labeled video. In *CVPR*.

Tang, S., Andriluka, M., Andres, B., and Schiele, B. (2017). Multiple people tracking by lifted multicut and person re-identification. In *CVPR*.

Tomasello, M., Carpenter, M., and Liszkowski, U. (2007). A new look at infant pointing. *Child development*.

Torralba, A., Oliva, A., Castelhano, M., and Henderson, J. M. (2006). Contextual guidance of attention in natural scenes: The role of global features on object search. *Psychological Review*, 113(4):766–786.

Tuytelaars, T. and Van Gool, L. (2004). Matching widely separated views based on affine invariant regions. *IJCV*, 59(1):61–85.

Uijlings, J., Popov, S., and Ferrari, V. (2017). Revisiting knowledge transfer for training object class detectors. *arXiv preprint arXiv:1708.06128*.

Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., and Smeulders, A. W. M. (2013). Selective search for object recognition. *IJCV*.

Van de Sande, K., J.R.R., U., Gevers, T., and Smeulders, A. (2011). Segmentation as selective search for object recognition. In *ICCV*.

Vedaldi, A. and Fulkerson, B. (2008). VLFeat: An open and portable library of computer vision algorithms.

Vedaldi, A., Gulshan, V., Varma, M., and Zisserman, A. (2009). Multiple kernels for object detection. In *ICCV*.

Veksler, O. (2008). Star shape prior for graph-cut image segmentation.

Veksler, O., Boykov, Y., and Mehrani, P. (2010). Superpixels and supervoxels in an energy optimization framework. In *ECCV*, pages 211–224.

Vezhnevets, A. and Ferrari, V. (2014). Associative embeddings for large-scale knowledge transfer with self-assessment. In *CVPR*.

Vicente, S., Kolmogorov, V., and Rother, C. (2008). Graph cut based image segmentation with connectivity priors. In *CVPR*.

Vig, E., Dorr, M., and Cox, D. (2012). Saliency-based space-variant descriptor sampling for action recognition. In *ECCV*.

Vijayanarasimhan, S. and Grauman, K. (2008). Multi-level active prediction of useful image annotations for recognition. In *NIPS*.

Vijayanarasimhan, S. and Grauman, K. (2009). What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations. In *CVPR*.

Vijayanarasimhan, S. and Grauman, K. (2014). Large-scale live active learning: Training object detectors with crawled data and crowds. *IJCV*, 108(1-2):97–114.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages 511–518.

Viola, P. A., Platt, J., and Zhang, C. (2005). Multiple instance boosting for object detection. In *NIPS*.

Vondrick, C., Patterson, D., and Ramanan, D. (2013). Efficiently scaling up crowdsourced video annotation. *IJCV*.

Wah, C., Branson, S., Perona, P., and Belongie, S. (2011). Multiclass recognition and part localization with humans in the loop. In *ICCV*.

Wah, C., Van Horn, G., Branson, S., Maji, S., Perona, P., and Belongie, S. (2014). Similarity comparisons for interactive fine-grained categorization. In *CVPR*.

Walber, T., Scherp, A., and Staab, S. (2013). Can you see it? two novel eye-tracking-based measures for assigning tags to image regions. In *MMM*.

Wang, C., Ren, W., Zhang, J., Huang, K., and Maybank, S. (2015). Large-scale weakly supervised object localization via latent category learning. *IEEE Transactions on Image Processing*, 24(4):1371–1385.

Wang, J. and Cohen, M. (2005). An iterative optimization approach for unified image segmentation and matting. In *ICCV*.

Wang, L., Hua, G., Sukthankar, R., Xue, J., and Zheng, J. (2014a). Video object discovery and co-segmentation with extremely weak supervision. In *ECCV*.

Wang, T., Han, B., and Collomosse, J. (2014b). Touchcut: Fast image and video segmentation using single-touch interaction. *CVIU*.

Wang, X., Yang, M., Zhu, S., and Lin, Y. (2013). Regionlets for generic object detection. In *ICCV*, pages 17–24. IEEE.

Welinder, P., Branson, S., Perona, P., and Belongie, S. J. (2010). The multidimensional wisdom of crowds. In *NIPS*.

Wolfe, J. and Horowitz, T. S. (2008). Visual search. *Scholarpedia*, 3(7):3325.

Wood, E., Baltrusaitis, T., Zhang, X., Sugano, Y., Robinson, P., and Bulling, A. (2015). Rendering of eyes for eye-shape registration and gaze estimation. In *CVPR*.

Wu, J., Zhao, Y., Zhu, J.-Y., Luo, S., and Tu, Z. (2014). Milcut: A sweeping line multiple instance learning paradigm for interactive image segmentation. In *CVPR*.

Xiao, J., Xu, P., Zhang, Y., Ehinger, K., Finkelstein, A., and Kulkarni, S. (2015). What can we learn from eye tracking data on 20,000 images? *Journal of vision*, 15(12):790–790.

Xu, P., Ehinger, K. A., Zhang, Y., Finkelstein, A., Kulkarni, S. R., and Xiao, J. (2015). Turkergaze: Crowdsourcing saliency with webcam based eye tracking. *arXiv preprint arXiv:1504.06755*.

Yang, W., Cai, J., Zheng, J., and Luo, J. (2010). User-friendly interactive image segmentation through unified combinatorial user inputs. *IEEE Transactions on Image Processing*, 19(9):2470–2479.

Yao, A., Gall, J., Leistner, C., and Van Gool, L. (2012). Interactive object detection. In *CVPR*.

Yun, K., Peng, Y., Samaras, D., Zelinsky, G. J., and Berg, T. L. (2013). Studying relationships between human gaze, description, and computer vision. In *CVPR*.

Zhang, J., Marszałek, M., Lazebnik, S., and Schmid, C. (2007). Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 73(2):213–238.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. (2016). Learning deep features for discriminative localization. In *CVPR*, pages 2921–2929.

Zitnick, C. L. and Dollár, P. (2014). Edge boxes: Locating object proposals from edges. In *ECCV*.